Identifiability and Observability Assessment for Nonlinear Wind Turbine Control Systems

Thomas Schmitt

Masterthesis – 30. November 2017

Supervisor: Bastian Ritter, M.Sc.

TECHNISCHE
UNIVERSITÄT
DARMSTADT

REGELUNGSTECHNIK
UND MECHATRONIK rtm

Problem Definition

After three decades of intense research, wind energy plays a significant role in energy production world wide and continues to replace fossil fuels. The major part of wind energy is allocated by wind turbines with an electrical generator and horizontal axis. This work is concerned with the physical modeling and state estimation of these plants as a basis for modern state space control concepts. However, from a practical point of view it is essential to know the conditions for good observability of the states and identifiability of the parameters. For linear systems there are different analysis tools available. However, due to aerodynamics, wind turbines can only be well described for the entire working range by nonlinear design models. Therefore, in this thesis, possibilities for the assessment of observability shall be investigated and applied. The results are to be documented in a detailed elaboration and discussed critically.

Begin: 01. Juni 2017
End: 30. November 2017

Prof. Dr.-Ing. Ulrich Konigorski

Bastian Ritter, M.Sc.

Technische Universität Darmstadt
Institut für Automatisierungstechnik und Mechatronik
Fachgebiet Regelungstechnik und Mechatronik
Prof. Dr.-Ing. Ulrich Konigorski

Landgraf-Georg-Straße 4
64283 Darmstadt
Telefon 06151/16-4167
www.rtm.tu-darmstadt.de

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 30. November 2017

Thomas Schmitt

Abstract

This work investigates the identifiability and observability of two nonlinear wind turbine models. Various definitions of both concepts are summarized together with an interpretation of their relation. An overview of existing methods to assess the identifiability and observability of nonlinear systems qualitatively as well as quantitatively is given. Of these, the profile likelihood approach is chosen and applied to both models. Thereby, statistical confidence intervals for parameters and states are derived. The identifiability of the air density, eigenfrequency and wind velocity as well as the observability of all states is assessed for various wind scenarios and measurement configurations. A qualitative overview is given together with a detailed analysis for selected constellations. Furthermore, the validity of the used methodology is verified.

Keywords: profile Likelihood, prediction, PL, PPL, Data2Dynamics, D2D, confidence intervals, nonlinear systems

Kurzfassung

Diese Arbeit analysiert die Identifizierbarkeit und Beobachtbarkeit zweier nichtlinearer Windturbinenmodelle. Verschiedene Definitionen beider Konzepte werden zusammen mit einer Interpretation ihrer Beziehung zueinander zusammengefasst. Außerdem wird ein Überblick existierender Methoden gegeben, mit denen die Identifizierbarkeit und Beobachtbarkeit nichtlinearer Systeme sowohl qualitativ als auch quantitativ bewertet werden kann. Von diesen wurde der Profile Likelihood Ansatz ausgewählt und auf beide Modelle angewendet. Dabei werden statistische Konfidenzintervalle für Parameter und Zustände abgeleitet. Die Identifizierbarkeit der Luftdichte, Eigenfrequenz und Windgeschwindigkeit sowie die Beobachtbarkeit aller Zustände werden für verschiedene Windszenarien und Messkonfigurationen bewertet. Diese sind sowohl in einem qualitativen Überblick zusammengefasst als auch für ausgewählte Konstellationen detailliert analysiert. Darüber hinaus wird die Validität der verwendeten Methodik verifiziert.

Contents

Symbols and Abbreviations

General Connotation

Symbol	Description
$\boldsymbol{x}$	bold lowercase letter, vector
$\boldsymbol{A}$	bold uppercase letter, matrix
$\dot{x}$	dotted variable, time derivative
$\widetilde{x}_{\mathrm{T}}$	tilde, absolute estimation error of a state
$x_{\mathrm{T},0}$	zero in subscript, initial value

General Symbols and Formula Symbols

Symbol	Description
y	output
x	state
u	input
θ	unknown parameter
σ	measurement error
ϵ	measurement noise
χ^2	objective function of the profile likelihood approach
α	significance level
Δ_α	α-quantile of the χ^2-distribution
df	degree of freedoms
$\#\boldsymbol{\theta}$	number of parameters
$\mathrm{L}(\boldsymbol{y}\|\boldsymbol{\theta})$	likelihood function
$\boldsymbol{\theta}^*$	maximum likelihood estimates
ι	wind turbulence
q	averaged sum of the relative errors
r	Pearson correlation coefficient
r^2	coefficient of determination

Wind Turbine Models Quantities and Parameters

Symbol	Description	MATLAB name
x_{T}	nacelle fore-aft position	`TTDspFA`
$\dot{x}_{\mathrm{T}}$	nacelle fore-aft velocity	`NcIMUTVxs`

Symbol	Description	MATLAB name
$\ddot{x}_{\mathrm{T}}$	nacelle fore-aft acc.	`NcIMUTAxs`
y_{T}	nacelle side-side position	`TTDspSS`
$\dot{y}_{\mathrm{T}}$	nacelle side-side velocity	`NcIMUTVys`
$\ddot{y}_{\mathrm{T}}$	nacelle side-side acc.	`NcIMUTAys`
φ_{g}	generator azimuth angle	`Q_GeAz`
$\dot{\varphi}_{\mathrm{g}}$	generator angular speed	`QD_GeAz`
$\ddot{\varphi}_{\mathrm{g}}$	generator angular acc.	`QD2_GeAz`
$\Delta\varphi$	drive-train torsion	`Q_DrTr`
$\Delta\dot{\varphi}$	drive-train angular speed	`QD_DrTr`
$\Delta\ddot{\varphi}$	drive-train angular acc.	`QD2_DrTr`
n_{g}	generator speed	`GenSpeed`
M_{g}	generator torque	`GenTq`
β_i	blade pitch	`BldPitch1`, `BldPitch2`, `BldPitch3`
v_{w}	hub-height wind speed	`Wind1VelX`
v_b	blade effect. wind speed	
ψ_b	blade azimuth angle	
λ_b	blade tip speed ratio	
ϱ	air mass density	`AirDensity`
C_{M}	torque coefficient	
C_{T}	thrust coefficient	
$c_{\mathrm{m,i}}$	polynomial coefficient for C_{T}	`cm0`, `cm1`, `cm2`
$c_{\mathrm{t,i}}$	polynomial coefficient for C_{M}	`ct0`, `ct1`, `ct2`
R	blade tip radius	`BldTpRd`
i_{gb}	drive-train gear-box ratio	`GbRatio`
H	hub height	`HubHeight`
κ	vertical shear exponent	`VerShr`
ω_0	eigenfrequency	`Omega0X`
m_{T}	equiv. tower top mass	`EqTwrMass`
Θ_r	combined rotor and blade inertia	
Θ_g	generator low-speed inertia	
$k_{\mathrm{T}x}$	equiv. fore-aft stiffness	`TwrStfnFA`
$k_{\mathrm{T}y}$	equiv. side-side stiffness	`TwrStfnSS`
$b_{\mathrm{T}x}$	equiv. fore-aft damping coeff.	`TwrDmpFA`
$b_{\mathrm{T}y}$	equiv. side-side damping coeff.	`TwrDmpSS`
k_φ	torsional stiffness coeff.	`DtTorStfn`
b_φ	torsional damping coeff.	`DtTorDmp`
ζ	beam coupling coeff.	`BeamCoup1`
$\zeta_{\mathrm{T}x}$	fore-aft damping coeff.	`zetaX`
$\zeta_{\mathrm{T}y}$	side-side damping coeff.	`zetaY`

Symbol	Description	MATLAB name
r_n	effect. normal radius	`NormEffRd1`
r_t	effect. tangential radius	`TangEffRd1`
r_B	power effect. radius	`PwrEffRd`

Abbreviations

Acronym	Complete Denomination
ODE	ordinary differential equation
SD	standard deviation
D2D	Data2Dynamics
PL	profile likelihood
PLE	profile likelihood exploit
PPL	prediction profile likelihood
CI	confidence interval
PCI	prediction confidence interval
VCI	validation confidence interval
SNR	signal-to-noise ratio
MCMC	Marcov chain Monte Carlo

1 Introduction

This chapter first explains the motivation behind this work. Then, the limits of its scope are set. Lastly, a description of its structure gives the reader an overview of what to expect.

1.1 Motivation

One of the biggest problems humanity is facing today is climate change. The reasons for this increase in global temperature are well known, and include a rising population, an increased average energy use per capita and a boost in emissions. The most promising countermeasure is an extensive energy transition to renewable energy sources. A big part of these sources are wind turbines. Using the wind to serve the needs of humanity is an ancient practice. The first Egyptian sailing ships are recorded from approximately 3000 BC, and grinding windmills were built as early as 900 AD. Charles Brush was the first person who used wind energy to produce electricity in 1888. Since then, many concepts have been tested, with successes and failures on the way.

However, today we have come to a point where wind turbines are not only used extensively, but also have the potential to be the prime source of energy in the future. Historically, the most significant research about wind turbines has been conducted in Germany, Denmark and the USA, where wind energy is already playing a big role in renewable energies. Furthermore, especially China with its huge increase in energy demand is investing massively in wind energy, and has plans to spend $ 100 Billion USD over the next four years.

Although over the years different shapes and forms of wind turbines have been built and tested,[1] today, the three bladed upwind horizontal axis wind turbine (see Figure 1.1) is the nearly exclusively used model. Until now, researchers have successfully worked on optimizing the mechanical structure of a wind turbine, i. e. we know the theoretically optimal blade form and can build foundations which withhold storms. We can also model and simulate them very accurately, at the latest since the National Renewable Energy Laboratory (NREL) of the U.S. Department of Energy has developed and published the FAST simulator.[2]

However, in terms of controlling wind turbines, especially as wind farms, there are still challenges to face. In general, a wind turbine can only be represented accurately by a nonlinear model, which makes developing controllers much more complex. Furthermore, the more advanced controllers one wants to use, the more information about the system's internals must be known, especially its dynamic states for any type of state space controller. At the same time,

[1] See [1] for an introduction in the history of wind turbines.

[2] `https://nwtc.nrel.gov/FAST`

Figure 1.1.: Typical modern wind turbine in a farm. Source: [1]

parameters of the wind turbine models, such as the air density and the eigenfrequency, may vary over time. Others such as the wind velocity additionally vary over height and are hard to measure. Therefore, an important part of developing controllers for wind turbines is observing the system's states and identifying parameters such as mentioned above. Usually, observers like the various types of Kalman filters are used for this task. However, before an observer can be designed for a system, the control engineer needs to ensure that the system is observable (and/or identifiable) at all - and not only for a specific operating point.

For linear systems, the theory of a system's observability and identifiability is practically completely discussed and also relatively easy to apply. For nonlinear models, there are analytical theorems to determine whether a system is observable/identifiable or not, too. However, with more advanced models, most of the times they are too complex to apply - at least if complex types of inputs are assumed. Therefore, other approaches must be found and applied to wind turbine models to enhance further possibilities for their control - that is where this work ties in.

1.2 Scope of Work

This work covers the identifiability and observability assessment of nonlinear systems defined by ordinary differential equations (ODEs). At the beginning, a thorough literature review about the general identifiability and observability theory of nonlinear systems has been conducted, especially with a view to methods possibly applicable to wind turbine models. Various methods for both identifiability and observability assessment have been found of which the most interesting are summarized here. Depending on their complexity and their relevance for the latter part of this work, they are discussed in more or less detail. Some are not discussed at all but recorded to give a complete overview as possible. Furthermore, for the first time a classification of available methods in dependence on their result, i. e. qualitative or quantitative, and their methodology, i. e. whether they use only model or also data information, is given together with the overview.

Of all these available methods, the most promising has been elected: the Profile Likelihood (PL) approach. It can be applied to assess both a systems identifiability and observability qualitatively as well as quantitatively. Furthermore, a framework for MATLAB, called *Data2Dynamics* (D2D), is available. Since until now this framework has mostly been used for biological systems, a lot of implementation effort was necessary. Thus, its application with mechanical models as ours is described here, too.

As wind turbine models we use two nonlinear state space models of different complexity, one with three and one with eight states, which have been previously developed. Their derivation is not part of this works scope, but their implications on the identifiability and observability analysis is. Moreover, the results of these analyses for both models are given, i. e., a qualitative overview is presented under which circumstances these models are identifiable/observable, as well as detailed quantitative examinations for various scenarios.

Lastly, we validate our methodology. For the identifiability assessment, we can compare our identifiability measure directly to the accuracy of the parameter estimates found in the process. With the methodology chosen in this work, we do net only get estimates for the parameters, but also for the states. Thus, we can also compare our observability measure with the states' estimation errors.

However, it must be noticed that the design of observers, filters or other applications to identify or observe any quantities online is not part of this work.

1.3 Thesis Structure

After this introduction, the thesis continues with the current state of research regarding the identifiability and observability assessment of nonlinear ODE systems in chapter 2. Thereby, in Section 2.1, definitions and explanations of both are stated, together with an interpretation of the relation between these concepts. Then, in Section 2.2, a categorized overview of all found methods is given in the form of a table. Subsequently, some of these methods are discussed, again categorized by their purpose (identifiability/observability assessment and qualitative/quantitative results).

Chapter 3 covers both modeling and methodology. First, our two used nonlinear ODE models of a wind turbine are presented in Section 3.1. In there, all necessary equations and nomenclature are defined. Following, in Section 3.2 the MATLAB framework *Data2Dynamics* (D2D) is first explained in principle. Then, the implementation of our models and data sets of them in D2D is described in Section 3.3. Lastly, to complement our methodology, possible scenario variations for our analyses are discussed in Section 3.4.

Chapter 4 deals with the simulation results obtained from applying the profile likelihood approach to our models in D2D. First, the influence of data quality and quantity is discussed in Section 4.1, mainly to decrease the number of factors we need to evaluate. Then, qualitative overviews of the identifiability and observability of different constellations for both models are

given in Section 4.2. In the following, quantitative and more detailed reflections of selected constellations regarding their identifiability (Section 4.3) and their observability (Section 4.4) are discussed. We evaluate the validity of the profile likelihood approach's application to our models in Section 4.5.

Finally, Chapter 5 summarizes this work's results in Section 5.1, followed by a conclusion in Section 5.2. An outlook for future work as a follow-up of this thesis is given in Section 5.3.

2 Current State of Research

This chapter gives an overview of the necessary theoretical background to assess the identifiability and observability of a nonlinear system. Therefore, several definitions of and a differentiation between these two concepts are presented first. Then, an overview of currently available methods to test for or assess identifiability and observability is given, whereas the methods are differentiated by their methodology as well as their outcome.

2.1 Identifiability and Observability of Nonlinear Systems

As the aim of this work is to analyze nonlinear models of wind turbine systems, a general definition of a nonlinear system is needed. We restrict the nonlinearities to ordinary differential equations (ODEs) described by

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\theta}), \tag{2.1}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{\theta}), \tag{2.2}$$

even though some methods described later on might refer to slightly varied model descriptions. Hereby, $\boldsymbol{x}(t) \in \mathbb{R}^n$ is the state vector, $\boldsymbol{u}(t) \in \mathbb{R}^m$ the input vector, $\boldsymbol{y}(t) \in \mathbb{R}^r$ the output vector and $\boldsymbol{\theta}$ a vector containing all unknown scalar parameters and possibly also unknown starting conditions.

2.1.1 Definitions of Identifiability

There are several definitions of different types of identifiability. First, we give a general definition adapted from [2].

Definition 2.1 *A system described by Eqs.* (2.1) *and* (2.2) *is* identifiable*, if $\boldsymbol{\theta}$ can be determined from the given, suitable system input $\boldsymbol{u}(t)$ and the measurable system output $\boldsymbol{y}(t)$.*

Albeit being a system property, identifiability requires a *suitable* input, which means that the input must excite the system sufficiently. This requirement was formalized by L. Ljung in [3], [4], giving a definition for an input to be *wide-sense persistently exciting*.

Furthermore, it can be distinguished between *global* and *local* identifiability, introduced in [4].

Definition 2.2 *A system as described by Eqs.* (2.1) *and* (2.2) *is* globally identifiable*, if for any suitable input $\boldsymbol{u}(t)$, $\boldsymbol{y}(\boldsymbol{u}, \boldsymbol{\theta}_1) = \boldsymbol{y}(\boldsymbol{u}, \boldsymbol{\theta}_2)$ holds if and only if $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$ for any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$ in the parameter space Θ.*

Definition 2.3 *A system as described by Eqs.* (2.1) *and* (2.2) *is* locally identifiable*, if for any suitable input* $\boldsymbol{u}(t)$ *and any* $\boldsymbol{\theta}_1$ *within an open neighborhood of* $\boldsymbol{\theta}_2$*,* $\boldsymbol{y}(\boldsymbol{u}, \boldsymbol{\theta}_1) = \boldsymbol{y}(\boldsymbol{u}, \boldsymbol{\theta}_2)$ *holds if and only if* $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$*.*

Another important distinction is between *structural* (also called *a priori*[1]) and *practical* identifiability. Since at least practical identifiability cannot be defined universally, formal definitions for these two categories will be introduced later on within methods for their assessment. However, it shall be noticed that structural identifiability refers to the theoretical possibility to identify the parameters $\boldsymbol{\theta}$ from $\boldsymbol{y}$ and $\boldsymbol{u}$ as stated in the definitions above, not respecting any limitations as quantity or quality of the available data, meaning it is a qualitative 'yes or no'-concept.

Practical identifiability considers limitations in fidelity and quantity of available data such as the amount of measurement points and noise. Therefore, its purpose is to gain a quantitative assessment of the identifiability of a system given practical circumstances. Note that structural identifiability is a necessary (but not sufficient) condition for practical identifiability.

It must be noted that not only the entire dynamic system, but also single parameters of $\boldsymbol{\theta}$ are considered to be identifiable or non-identifiable, i.e. some parameters may be identifiable while others are not. To illustrate how parameters may be non-identifiable, a small example from [5] shall be analyzed.

$$\dot{x}_1(t) = -\theta_1 x_1(t), \quad x_1(0) = \theta_1 \tag{2.3}$$
$$\dot{x}_2(t) = +\theta_1 x_1(t), \quad x_2(0) = 0 \tag{2.4}$$
$$y(t) = \theta_3 x_2(t)$$

Solving Eq. (2.3) analytically and substituting it into Eq. (2.4) yields in

$$\begin{aligned} x_1(t) &= \theta_2 e^{-\theta_1 t} \\ x_2(t) &= -\theta_2 (e^{-\theta_1 t} - 1) \\ y(t) &= \theta_3 \theta_2 e^{-\theta_1 t} \cdot (e^{\theta_1 t} - 1) \end{aligned} \tag{2.5}$$

As one can easily see in (2.5), the parameters θ_2 and θ_3 cannot be distinguished in the output y and are therefore (structural) non-identifiable.

In [2], *Miao et al.* summarize a statement of *Sontag* ([6]) which says that for an identifiable system with r parameters, $2r + 1$ measurement points are enough to determine $\boldsymbol{\theta}$. *Sontag* also indicated that additional measurements of a system in steady state do not add any new information to the identification problem. In contrast, data points during turbulent nonlinear behavior will be more informative for determining $\boldsymbol{\theta}$ ([7]).

[1] In this case, a priori refers to the theoretical identifiability before examining an experiment ([5], [2]). Since this term is also used in other concepts we will discuss, we solely refer to structural identifiability in the following.

2.1.2 Definitions of Observability

Defining observability of a nonlinear system is somewhat more complicated. Again, there are several definitions in the literature. Some of them are only applicable in combination with a certain method. If so, they will be given in Section 2.2.2, which discusses various methods to assess observability. Here, we concentrate on the two most popular and general definitions given in [8].

It must be noticed that, in contrast to linear systems, in the nonlinear case, observability highly depends on the input $\boldsymbol{u}(t)$. A nonlinear system may be observable for some $\boldsymbol{u}(t)$ and unobservable for others. The following two definitions refer to general observability of the system for all admissible inputs.

Definition 2.4 *A system as described by Eqs.* (2.1) *and* (2.2) *shall be defined for* $\boldsymbol{x} \in D_x \subseteq \mathbb{R}^n$ *and* $\boldsymbol{u} \in C_u \subseteq \mathbb{C}^{n-1}$ *with* $\boldsymbol{y} \in \mathbb{R}^r$. *If all starting vectors* $\boldsymbol{x}_0 \in D_x$ *can be uniquely determined from the knowledge of* $\boldsymbol{u}(t)$ *and* $\boldsymbol{y}(t)$ *in a finite time interval* $[t_0, t_1 < \infty]$ *for all* $\boldsymbol{u} \in C_u$, *the system is called* observable.

Definition 2.5 *A system as described by Eqs.* (2.1) *and* (2.2) *shall be defined for* $\boldsymbol{x} \in D_x \subseteq \mathbb{R}^n$ *and* $\boldsymbol{u} \in C_u \subseteq \mathbb{C}^{n-1}$ *with* $\boldsymbol{y} \in \mathbb{R}^r$. *If all starting vectors* $\boldsymbol{x}_0 \in D_x$ *in a neighborhood*

$$U = \{x_0 \in \mathbb{R}^n | \; ||x_0 - x_p|| < \rho\}$$

of a point $\boldsymbol{x}_p \in D_x$ *can be uniquely determined from the knowledge of* $\boldsymbol{u}(t)$ *and* $\boldsymbol{y}(t)$ *in a finite time interval* $[t_0, t_1 < \infty]$ *for all* $\boldsymbol{u} \in C_u$, *the system is called* weakly observable.

Notice that weak observability differs from *local observability*. Local observability at a point $\boldsymbol{x}_0$ would mean that a starting vector $\boldsymbol{x}_0$ could be determined uniquely in *any* open neighborhood of $\boldsymbol{x}_0$. Thus, local observability is a stronger concept then weak observability ([9]).

Here, an example from [8] shall be given, which illustrates how a dynamic system can be weakly observable, but not observable. Let the system be described by

$$\begin{aligned} \dot{x}(t) &= -\frac{1}{x}, \\ y &= x^2, \end{aligned}$$

with x defined on $D_x = \mathbb{R}\backslash\{0\}$. Obviously, the initial value x_0 cannot be determined uniquely from $y(t)$ but has two possible values,

$$x_{0,1} = -\sqrt{y(t_0)} \qquad \text{and} \qquad x_{0,2} = +\sqrt{y(t_0)} \; .$$

Hence, the system is not observable. However, in a suitable neighborhood $U = \{x_0 \in \mathbb{R}^n | \; ||x_0 - x_p|| < \rho\}$ for any point $x_p \in D_x$, the initial value can be determined, i. e.

$$\begin{aligned} x_0 &= -\sqrt{y(t_0)} \quad \text{for} \quad x_p < 0, \quad \rho < |x_p| \; , \\ x_0 &= +\sqrt{y(t_0)} \quad \text{for} \quad x_p > 0, \quad \rho < |x_p| \; . \end{aligned}$$

Thus, the system is weakly observable.

2.1.3 Relation between Identifiability and Observability

Since now the concepts of identifiability and observability have been explained, a reflection of their relation is necessary. Both have in common that they reflect the possibility of gaining internal information of a system only from knowledge of the systems inputs and outputs. Also, both do require the model structure to be known. Hence, both rely on how well the output represents the internal structure (the ODEs) of the system. *Sedoglavic* [10] and *Krener and Ide* [11] pointed out that identifiability can even be interpreted as a special case of observability. Instead of identifying unknown scalar parameters, these parameters could be considered as a part of the state variable vector $\boldsymbol{x}_{\text{all}}$ without dynamics,

$$\dot{\boldsymbol{x}}_{\text{all}}(t) = \begin{pmatrix} \dot{\boldsymbol{x}}(t) \\ \dot{\boldsymbol{\theta}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{f}(\boldsymbol{x}_{\text{all}}(t), \boldsymbol{u}(t)) \\ 0 \end{pmatrix}, \tag{2.6}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}_{\text{all}}(t)). \tag{2.7}$$

Furthermore, the (real) state variables $\boldsymbol{x}$ usually depend on the parameters $\boldsymbol{\theta}$ through the ODEs. If some parameters are structurally non-identifiable, they cannot be distinguished in the output $\boldsymbol{y}$. Hence, a state variable x_i which depends on these parameters cannot be determined uniquely from the output $\boldsymbol{y}$, as was shown in [12]. That means that a state variable which depends on structural non-identifiable parameters cannot be observable.

As a different aspect, assuming a perfect model of the system with all ODE parameters known, the observation problem would be down to determining the initial conditions $\boldsymbol{x}_0$ of the state variables $\boldsymbol{x}$. Hence, in this case, with $\boldsymbol{\theta} = \boldsymbol{x}_0$ the observation problem could be considered a mere identification problem. Unfortunately, since in reality one can never expect a model to be completely correct, the observation of the state variables cannot entirely be disregarded to a one time identification of the initial conditions. However, we do assess a states' initial value's identifiability to at least qualitatively assess the states' observability in Section 4.2.2.

Concluding, every identification problem could be tackled as an observability problem and one might do so for convenience. However, assessing the identifiability instead of the observability of a nonlinear system can be easier in terms of mathematical comprehensibility, interpretation and computational effort. There are many methods which only cover identifiability assessments but offer system insights which more general methods might not. Therefore, in the following section, available methods for both identifiability and observability are represented.

2.2 Available methods ...

In this section, an overview of existing methods to assess both identifiability and observability of nonlinear systems is given. For linear systems, criteria for identifiability and especially observability are well known and are not treated particularly in this work to improve readability.

As explained in Section 2.1, the identifiability and observability of a system can not only be considered qualitatively (yes or no), but also quantitatively, i. e. how well a parameter θ_i (state variable x_i) can be identified (observed). Therefore, Table 2.1 summarizes available methods, distinguished by their

- result, i. e. qualitative or quantitative, and their
- methodology, i. e. whether only the model structure is used or (additionally) simulated and/or experimental data, too.

Table 2.1.: Overview of currently available methods to asses identifiability and observability.

Result	Methodology	Identifiability	Observability
Qualitative	Model only	• Power Series Expansion, see page 11 • Similarity Transformation, see page 12 • Direct Test, see page 12 • Differential Algebra, see page 13 • Implicit Function theorem (EAR), see page 13	• Standard algebraic observability theorems, see page 18
	Model + Data	• Profile Likelihood, see page 14	• Prediction Profile Likelihood, see page 27
Quantitative	Model only	• Several sensitivity based methods (Orthogonal method, Correlation method, PCA, Eigenvalue method), see [13] • Monte Carlo Simulations, Correlation Matrix (based on FIS), see [2]	• Empirical Gramian Matrix, see page 22 • Local unobservability index + local estimation condition number, see page 25 • (For LTI/LTP systems: distance to unobservability, see [14])
	Model + Data	• Profile Likelihood, see page 14 • Markov Chain Monte Carlo, see page 29	• Prediction Profile Likelihood, see page 27 • Markov Chain Monte Carlo, see page 29

In the following sections, some of these methods are discussed in depth. Others are only explained shortly and for some methods the reader is referred to the corresponding literature.

2.2.1 ... to assess Identifiability

First, methods to assess the identifiability of model parameters are discussed, starting with qualitative and followed by quantitative methods.

Qualitative Methods

Power Series Expansion, Similarity Transformation

Both the power series expansion (PSE) and the similarity transformation method include serious disadvantages. Therefore, none of them are used in this work and are presented only shortly.

In mathematics, a power series is in general an infinite series of the form

$$\sum_{n=0}^{\infty} a_n (x-c)^n = a_0 + a_1 (x-c)^1 + a_2 (x-c)^2 + \dots ,$$

which is used in various mathematical fields, e. g. as a solution approach for ODEs. *Pohjanpalo* used it in [15] to test nonlinear systems of the form

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{A}(t, \boldsymbol{x}, \boldsymbol{\theta})\boldsymbol{x} + \boldsymbol{u}, \\ \boldsymbol{y} &= \boldsymbol{C}(\boldsymbol{\theta})\boldsymbol{x}, \end{aligned}$$

for local identifiability. The system output $\boldsymbol{y}$ and its derivatives can be written as

$$\boldsymbol{y}(t_0) = \boldsymbol{C}\boldsymbol{x}(t_0) , \tag{2.8}$$

$$\boldsymbol{y}^k(t_0) = \boldsymbol{C}\left(\sum_{i=1}^{k} \frac{(k-1)!}{(k-i)!(i-1)!} \boldsymbol{A}^{(k-i)}(t_0)\boldsymbol{x}^{(i-1)}(t_0) + \boldsymbol{u}^{(k-1)}(t_0) \right) . \tag{2.9}$$

The derivatives $\boldsymbol{y}^{(k)}$ are theoretically observable ([2]). Thus, they are considered as known and infinitely many equations can be generated from Eqs. (2.8) and (2.9) to solve for $\boldsymbol{\theta}$. The system's (local) identifiability or non-identifiability depends on whether the solution is unique or not, respectively. However, the serious drawback of this method is its necessity for high order derivatives and the complexity of the resulting equations. Therefore, the PSE has never become popular in practice. But, it shall be noticed that another method, called the EAR approach, has been developed based on the PSE ([5]).

Similarity Transformation

The similarity transformation method (STM) was initially proposed by *Walter and Lecourtier* in [16] for linear systems,

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u},$$
$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x}.$$

Its aim is to obtain the similar matrix $\boldsymbol{S} = \boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P}$ of $\boldsymbol{A}$ such that

$$\dot{\boldsymbol{x}} = (\boldsymbol{P}^{-1}\boldsymbol{A}\boldsymbol{P})\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u},$$
$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x},$$

and lastly conclude global identifiability. *Vajda*, *Godfrey* and *Rabitz* ([17], [18]) extended the approach to non-linear SIMO systems of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta}) + u(t)\boldsymbol{h}(\boldsymbol{x}(t), \boldsymbol{\theta}),$$
$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{\theta}).$$

However, the approach can be applied to MIMO systems, too. Requirements for the application of the STM are that the system needs to be both controllable and observable and a set of partial differential equations must be generated and solved. Therefore, the STM is not a preferable method and will neither be applied or discussed further in this work.

Direct Test

The direct test proposed by *Denis-Vidal* and *Joly-Blanchard* [19] may be the most intuitive and comprehensible method to test a system for identifiability. To do so, the definition of identifiability is directly applied, i. e. we evaluate whether the implication

$$\boldsymbol{y}(\boldsymbol{u}, \boldsymbol{\theta}_1) = \boldsymbol{y}(\boldsymbol{u}, \boldsymbol{\theta}_2) \qquad \Rightarrow \qquad \boldsymbol{\theta}_1 = \boldsymbol{\theta}_2 \tag{2.10}$$

is true or not. For an uncontrolled, autonomous system the implication (2.10) is simplified to

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}_1) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta}_2) \qquad \Rightarrow \qquad \boldsymbol{\theta}_1 = \boldsymbol{\theta}_2.$$

Unfortunately, the mathematical solving of the equations becomes infeasible or even impossible very quickly since every unmeasured space variable x_i must be eliminated.

There are some frameworks for a numerical application of the direct test available. However, the user has to choose an arbitrary cut-off value δ and the result heavily depends on the chosen δ. Therefore, the conclusions are not reliable - a non-identifiable system may easily be considered identifiable ([2]). Furthermore, it is hard to distinguish which system parameters are identifiable and which are non-identifiable.

Differential Algebra

Since the algebraic handling of nonlinear systems can quickly become too complex to be done by hand, numerical approaches are frequently used, such as for the direct test described above. However, it is part of their nature that they can never be exact. Thus, a different approach is to let a computer do algebraic calculations, which is the goal of differential algebra in general. Shortly, ODEs are considered as differential polynomials and in dependency of their characteristics, e. g. the identifiability of a system as ours can be evaluated.

Since even the basics of differential algebra are complex enough, the reader is referred to [2] for an introduction and further literature. To use differential algebra to test a system for identifiability, the terms *algebraic identifiability* and *algebraic identifiability with known initial conditions* are defined. Since these definitions do not facilitate any comprehension of identifiability without the background of differential algebra, we refrain from recalling them.

Implicit Function Theorem

The theoretical background of the implicit function theorem (IFT) is the same as for the differential algebra. However, only regular algebra is used. The idea is to prove local identifiability at $\boldsymbol{\theta}_*$ by finding an identification function

$$\boldsymbol{\Phi} = \boldsymbol{\Phi}(\boldsymbol{\theta}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(k)}, \boldsymbol{y}, \dot{\boldsymbol{y}}, \ldots, \boldsymbol{y}^{(k)}),$$

for which

$$\boldsymbol{\Phi}(\boldsymbol{\theta}_*, \boldsymbol{u}_*, \dot{\boldsymbol{u}}_*, \ldots, \boldsymbol{u}_*^{(k)}, \boldsymbol{y}_*, \dot{\boldsymbol{y}}_*, \ldots, \boldsymbol{y}_*^{(k)}) = 0 \qquad \text{and} \qquad \left| \frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{\theta}_*} \right| \neq 0. \tag{2.11}$$

The conclusion of identifiability from Eq. (2.11) can easily be followed by consdering the Taylor expansion of $\boldsymbol{\Phi}$ at $\boldsymbol{\theta}_*$,

$$\boldsymbol{\Phi} \approx \boldsymbol{\Phi}(\boldsymbol{\theta}_*) + (\boldsymbol{\theta} - \boldsymbol{\theta}_*) \frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{\theta}_*},$$

which can directly be solved for $\boldsymbol{\theta}_*$ with $\boldsymbol{\Phi} = 0$ and $\left(\frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{\theta}_*}\right)^{-1}$, which exists since $\left|\frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{\theta}_*}\right| \neq 0$. Hence, the system is locally identifiable at $\boldsymbol{\theta}_*$.

Methods to find a function Φ are available ([20], [21]). In any case, the rank of the Jacobi matrix $\frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{\theta}_*}$ must be determined, which can become very complicated very quickly. Therefore, *Wu et al.* developed a less computational intensive method ([22]). For an introduction and a comparison example, the reader is referred to [2].

Profile Likelihood Approach

Raue et al. ([23], [12]) introduced a general approach to assess both structural and practical identifiability of nonlinear systems. In difference to our standard nonlinear system from Eqs. (2.1) and (2.2), they include measurement noise $\boldsymbol{\varepsilon}(t)$ on the output $\boldsymbol{y}$, leading to

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\theta}) \,, \tag{2.12}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}(t), \boldsymbol{\theta}) + \boldsymbol{\varepsilon}(t) \,. \tag{2.13}$$

To understand their approach, we need to recall some basics of parameter estimation. Its aim is to bring experimental data $\boldsymbol{y}^D$ in agreement with simulationally predicted data by minimizing an objective function, usually the weighted sum of squared residuals,

$$\chi^2(\boldsymbol{\theta}) = \sum_{k=1}^{r} \sum_{l=1}^{d} \left(\frac{y_{kl}^D - y_k(\boldsymbol{\theta}, t_l)}{\sigma_{kl}^D} \right)^2 . \tag{2.14}$$

Hereby, $y_k(\boldsymbol{\theta}, t_l)$ denotes the k-th of the r output variables resulting from a simulation of the system with parameters $\boldsymbol{\theta}$ at time t_l of all d data points and y_{kl}^D represents the measured data at the same point.

With (2.14), the parameters are estimated numerically as a minimization problem,

$$\hat{\boldsymbol{\theta}} = \arg\min \left[\chi^2(\boldsymbol{\theta}) \right].$$

Assuming normally distributed noise $\boldsymbol{\varepsilon}(t) \sim N(0,\sigma^2)$, χ^2 is proportional to the negative logarithm of the likelihood of $\boldsymbol{\theta}$ but a constant offset,[2]

$$(\chi^2 - \text{const}) \sim -\log(\text{L}(\boldsymbol{\theta})). \tag{2.15}$$

The aim of the profile likelihood approach is to derive confidence intervals for every parameter θ_i, i. e. deducing limits between which the parameter lies to a certain degree of likelihood. But, instead of trying to directly calculate the (log-)likelihood of the parameter estimate $\boldsymbol{\theta}$, with Eq. (2.15), we can compute the value of χ^2 for different values of a single parameter θ_i, called the profile likelihood (PL)

$$\chi^2_{\text{PL}}(\theta_i) = \min_{\theta_{j\neq i}} \left[\chi^2(\boldsymbol{\theta}) \right]. \tag{2.16}$$

Eq. (2.16) implies that we minimize Eq. (2.14) for a constant value of a parameter θ_i. Doing so over an interval of values for θ_i, the PL can be computed numerically. Figure 2.1 shows exemplary curves, of which (a) represents an identifiable parameter.

[2] The derivation of this relation is discussed in Section 3.3.2.

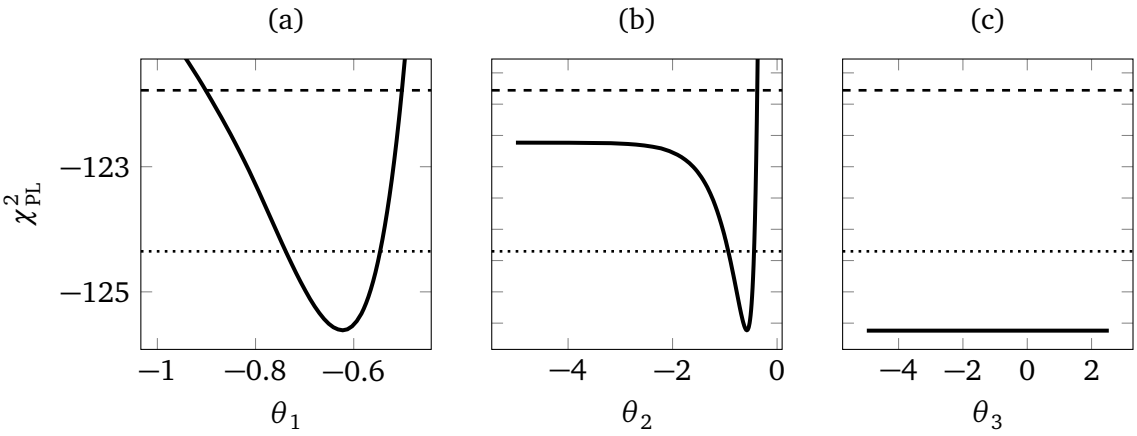

Figure 2.1.: Exemplary profile likelihood curves for three different parameters θ_1, θ_2 and θ_3. The lower dotted and the upper dashed lines indicate bounds for pointwise and simultaneous confidence intervals, respectively.

(a) structurally and practically identifiable parameter

(b) structurally, but not practically identifiable parameter

(c) parameter is neither practically nor structurally identifiable, i. e. non-identifiable.

As can be seen in (a), $\chi^2_{\mathrm{PL}}(\theta_1)$ has a minimum for $\theta_1 \approx 0{,}32$, i. e. this value has the highest probability to be true.

With PLs as in Figure 2.1, we can deduce confidence intervals for every parameter. More specific, we focus on *finite sample confidence intervals*[3], also called *likelihood-based confidence intervals*. They are derived by a threshold in the likelihood (or here in the PL) and defined by a confidence region

$$\{\boldsymbol{\theta} | \chi^2(\boldsymbol{\theta}) - \chi^2(\hat{\boldsymbol{\theta}}) < \Delta_\alpha\} \text{ with } \Delta_\alpha = \chi^2(\alpha, df) \,.$$

The threshold Δ_α is the α-quantile of the χ^2-distribution.[4] df represents the degree of freedom. With $df = 1$, the confidence interval is valid only pointwise, which means if all other parameters $\theta_{j\neq i}$ were true. With $df = \#\boldsymbol{\theta}$, the threshold for simultaneous confidence intervals for all parameters is calculated. Thus, both the threshold and the confidence interval are bigger for $df = \#\boldsymbol{\theta}$.

In Figure 2.1, the upper and lower dashed lines represent the thresholds for the simultaneous and the pointwise confidence intervals, respectively. As can be seen in (a), the parameter θ_1 has a unique minimum *and* bounded limits for both confidence intervals. Therefore, the parameter is both structurally and practically identifiable. Using the PL and likelihood-based confidence intervals, [23] introduces a definition for practical identifiability, recalled here as Definition 2.7.

[3] In [23], asymptotic confidence intervals are considered, too. However, asymptotic confidence intervals result in bounded limits even for non-identifiable parameters, which could lead to confusions, and do not yield any other advantage. Therefore, we restrain to finite sample confidence intervals.

[4] The χ^2-distribution does not coincide with (2.14). The nomenclature may be unfortunate, but has been adopted from [23] for better comparability.

Based on this approach, we introduce a formal definition of structural identifiability for completeness, too.

Definition 2.6 *A parameter θ_i is* structurally non-identifiable*, if its likelihood-based confidence interval is infinite, $[-\infty, +\infty]$, and its profile likelihood has no unique minimum, but is flat over all θ_i.*

Figure 2.1 (b) shows an example for a structurally non-identifiable parameter. Altering θ_2 does not effect $\chi^2_{\mathrm{PL}}(\theta_2)$, which means that θ_2 either has no impact on $\boldsymbol{y}$ at all, or there is a set of ambiguous parameters $\boldsymbol{\theta}_{\mathrm{sub}} \subset \boldsymbol{\theta}$, which may be altered without changing the output $\boldsymbol{y}$. This redundant parameterization can be expressed as functional relations $\boldsymbol{h}$ between the parameters $\boldsymbol{\theta}_{\mathrm{sub}}$, so that

$$\boldsymbol{h}(\boldsymbol{\theta}_{\mathrm{sub}}) = 0 \ .$$

Unfortunately, the PL approach does not include a general method to determine $\boldsymbol{h}$, but it is possible to determine the parameters of $\boldsymbol{\theta}_{\mathrm{sub}}$. To do so, we have to examine the course of the other parameters $\theta_{j \neq i}$ when calculating $\chi^2_{\mathrm{PL}}(\theta_i)$ for different values of a structurally non-identifiable parameter θ_i, as is exemplary shown in Figure 2.2.

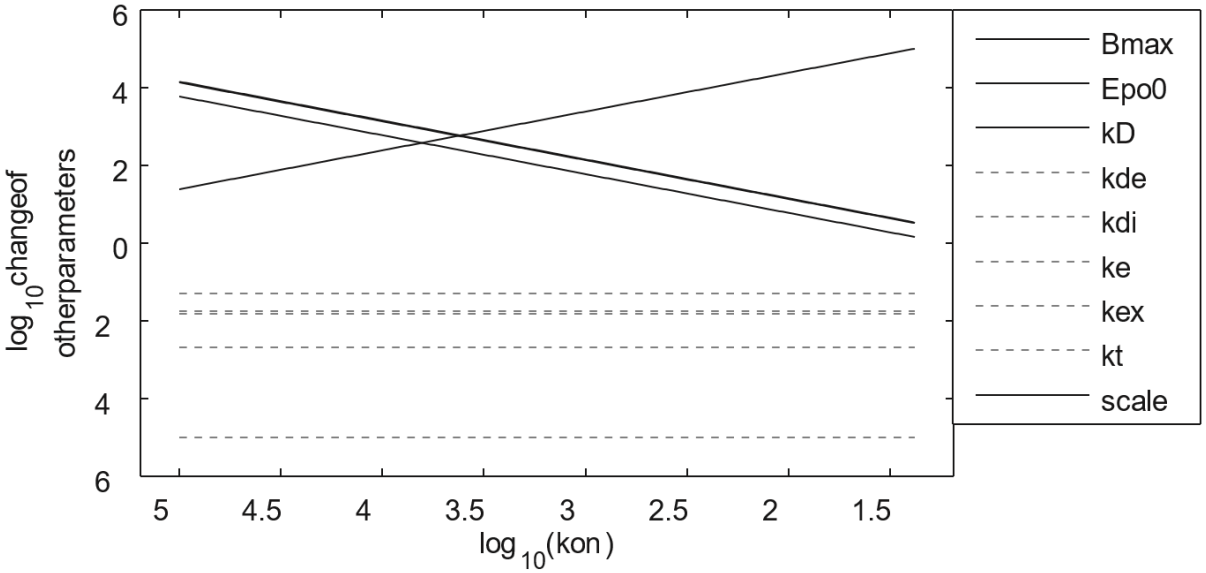

Figure 2.2.: Example of other parameters $\theta_{j \neq i}$ developments when the profile likelihood is calculated for one parameter, e. g. for $\theta_i = \mathrm{kon}$ here. Source: [12]

As can be seen, the parameters Bmax, Epo0 and kD change linearly with kon (note that both x- and y-axis are in a $\log_{10}$ scale). Hence, these four parameters form $\boldsymbol{\theta}_{\mathrm{sub}}$. However, there could be one or more other sets of structurally non-identifiable parameters in $\boldsymbol{\theta}_{\mathrm{sub}}$, which can be spotted by plotting the PL for every parameter θ_i.

If the corresponding manifold is only one-dimensional, i. e. one of the parameters changes within the PL of only one different parameter, the functional relation between these parameters can be derived quantitatively, e. g. in form of

$$\boldsymbol{h}_1(\boldsymbol{\theta}_{\text{sub}}) = \theta_1 \cdot \theta_2 - \text{const} = 0 \; .$$

For higher dimensional manifolds, the reader is referred to [24] which introduces a non-parametric bootstrap-based method for identifiability testing.

Now, the formal definition of practical non-identifiability proposed by [23] shall be cited.[5]

Definition 2.7 *A parameter θ_i is* practically non-identifiable*, if the likelihood-based confidence region is infinitely extended in increasing and/or decreasing direction of θ_i, although the likelihood has a unique minimum for this parameter.*

Figure 2.1 (c) shows an example for a practically non-identifiable parameter. The simultaneous confidence interval of θ_3 is finite only to one end,[6] even though it does have a unique minimum. Thus, to successfully identify the systems parameters, the practical non-identifiabilities need to be resolved by an improved measurement, which may involve

- the measurement of additional observables, i. e. increasing the output vector $\boldsymbol{y}$,
- more measurement points and/or
- decreasing measurement noise, i. e. a decreased variance of $\boldsymbol{\varepsilon}(t)$.

To evaluate the effects of an uncertain parameter θ_i, whether through a practically/structurally non-identifiability or just a big confidence interval, the system can be simulated along its PL $\chi^2(\theta_i)$, i. e. for all the resulting parameters $\theta_{j\neq i}$ during the calculation of $\chi^2(\theta_i)$. In addition, the plots of the state variables $\boldsymbol{x}$ and output variables $\boldsymbol{y}$ can be considered to improve experimental planning. If a unmeasured state variable x_i is varying strongly over $\chi^2(\theta_i)$, it indicates that including x_i in $\boldsymbol{y}$ could resolve the identification problem of θ_i. If the simulated value of a measured output variable y is varying strongly only over a period of time, additional measurement points during this period could help, too. By plotting the system's variables $\boldsymbol{x}$ over the PLs of the parameters, we do also gain insights on the observability of the system. However, to gain a reliable assessment of a system's observability by using the PL methodology, *Kreutz et al.* suggested a continuative approach in [25], which is called *prediction profile likelihood* and described in Section 2.2.2 on page 27.

[5] In [23], the definition refers to a 'parameter estimate $\hat{\theta}_i$' to be practically non-identifiable. Since identifiability is a system property, we formulate it in behalf of the system 'parameter θ_i'. Otherwise, the definition is taken word for word from [23].

[6] The pointwise confidence interval would be finite to both ends. However, since we cannot assume the other parameters to be known, only the simultaneous confidence interval is valid. Therefore, in an experiment, θ_3 would be practically non-identifiable.

2.2.2 ... to assess Observability

After various methods to assess a nonlinear system's identifiability have been presented, the same is done for its observability. Again, first qualitative methods are discussed followed by methods with quantitative results.

Qualitative Methods

Standard Algebraic Theorems

To give a smooth introduction into the topic of observability, we shall recall the well-known observability theorem for linear time-invariant (LTI) systems, e. g. given in [26] and [27].

Theorem 2.8 *The linear system*

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{2.17}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x}, \tag{2.18}$$

with $\boldsymbol{x} \in R^n$*, is* observable*, iff the* observability matrix of the second kind*,*

$$\overline{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{C}\boldsymbol{A} \\ \boldsymbol{C}\boldsymbol{A}^2 \\ \vdots \\ \boldsymbol{C}\boldsymbol{A}^{n-1} \end{bmatrix},$$

has the rank n.

Using the observability matrix of the second kind, $\overline{\boldsymbol{M}}$, the system output and its derivatives can be written as

$$\begin{bmatrix} \boldsymbol{y}(t) \\ \dot{\boldsymbol{y}}(t) \\ \vdots \\ \boldsymbol{y}^{(n-1)}(t) \end{bmatrix} = \overline{\boldsymbol{M}}\boldsymbol{x}(t).$$

Apparently, the observability of a system depends on whether or not the output $\boldsymbol{y}(t)$ and its derivatives contain enough information about the state variables. Following this idea and introducing the operator[7]

$$F_A\{\cdot\} = \cdot\, A(t) + \frac{\mathrm{d}}{\mathrm{d}x}\,\cdot\ ,$$

a similar theorem for linear time-variant (LTV) systems can be given, too. For this, we define the *observability matrix of the first kind* $\overline{\boldsymbol{M}}(t)$ [8] as

$$\begin{bmatrix} \boldsymbol{y}(t) \\ \dot{\boldsymbol{y}}(t) \\ \ddot{\boldsymbol{y}}(t) \\ \vdots \\ \boldsymbol{y}^{(n-1)}(t) \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}(t) \\ F_A\{\boldsymbol{C}(t)\} \\ F_A^2\{\boldsymbol{C}(t)\} \\ \vdots \\ F_A^{n-1}\{\boldsymbol{C}(t)\} \end{bmatrix} \boldsymbol{x}(t) = \overline{\boldsymbol{M}}(t)\,\boldsymbol{x}(t)\,. \tag{2.19}$$

According to [27], we can now deduce the following theorem.

Theorem 2.9 *Assume the time-variant version of the linear system in Eqs.* (2.17) *and* (2.18) *with* $\boldsymbol{A}(t)$, $\boldsymbol{B}(t)$ *and* $\boldsymbol{C}(t)$. *The system is* observable*, iff the observability matrix of the first kind* $\overline{\boldsymbol{M}}(t)$ *in Eq.* (2.19) *has the rank n.*

For nonlinear systems, their observability can be deduced from an expression such as in (2.19), too. For simplicity, we first consider nonlinear MISO systems as in [8], but without a direct feedthrough, given by

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})\,, \tag{2.20}$$

$$y(t) = g(\boldsymbol{x})\,. \tag{2.21}$$

Then, the $(n-1)$ derivatives of y are

$$\begin{aligned} \dot{y} &= \frac{\partial g}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) = h_1(\boldsymbol{x}, \boldsymbol{u})\,, \\ \ddot{y} &= \frac{\partial h_1}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \frac{\partial h_1}{\partial \boldsymbol{u}} \dot{\boldsymbol{u}} = h_2(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}})\,, \\ \dddot{y} &= \frac{\partial h_2}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \frac{\partial h_2}{\partial \boldsymbol{u}} \dot{\boldsymbol{u}} + \frac{\partial h_2}{\partial \dot{\boldsymbol{u}}} \ddot{\boldsymbol{u}} = h_3(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ddot{\boldsymbol{u}})\,, \\ &\vdots \\ y^{(n-1)} &= \frac{\partial h_{n-2}}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) + \sum_{i=1}^{n-2} \frac{\partial h_{n-2}}{\partial \boldsymbol{u}^{(i-1)}} \boldsymbol{u}^{(i)} = h_{(n-1)}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right)\,, \end{aligned}$$

and we summarize them as a mapping

$$\boldsymbol{z} = \begin{bmatrix} y \\ \dot{y} \\ \ddot{y} \\ \vdots \\ y^{(n-1)} \end{bmatrix} = \begin{bmatrix} g(\boldsymbol{x}) \\ h_1(\boldsymbol{x}, \boldsymbol{u}) \\ h_2(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}) \\ \vdots \\ h_{(n-1)}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right) \end{bmatrix} = \boldsymbol{q}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right)\,.$$

Again, z determines the observability of a system. The following theorems can be deduced (see e. g. [8]).

[7] In [27], $L_A\{\cdot\}$ is used for this operator. Since the operator is similar to the Lie derivative used later on for the nonlinear observability theorem from [8] and might be mistaken, it was changed to $F_A\{\cdot\}$.

[8] Note that the denotation of the observability matrix of the first and second kind, $\overline{\boldsymbol{M}}(t)$ and $\overline{\boldsymbol{M}}$, respectively, only differ in the time dependence (t).

Theorem 2.10 *A nonlinear MISO system described by Eqs.* (2.20) *and* (2.21) *and defined for* $\boldsymbol{x} \in D_x \subseteq \mathbb{R}^n$ *and* $\boldsymbol{u} \in C_u \subseteq \mathbb{C}^{n-1}$ *is* observable*, if the mapping*

$$\boldsymbol{z} = \boldsymbol{q}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right)$$

is uniquely invertible with respect to $\boldsymbol{x}$.

However, the application of Theorem 2.10 can be very complicated or even impossible. Therefore, a more applicable theorem is given.

Theorem 2.11 *A nonlinear MISO system described by Eqs.* (2.20) *and* (2.21) *and defined for* $\boldsymbol{x} \in D_x \subseteq \mathbb{R}^n$ *and* $\boldsymbol{u} \in C_u \subseteq \mathbb{C}^{n-1}$ *is* weakly observable*, if*

$$\operatorname{rank}\left(\frac{\partial \boldsymbol{q}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right)}{\partial \boldsymbol{x}}\right) = \operatorname{rank}\begin{bmatrix} \frac{\partial g(\boldsymbol{x})}{\partial \boldsymbol{x}} \\ \frac{\partial h_1(\boldsymbol{x},\boldsymbol{u})}{\partial \boldsymbol{x}} \\ \frac{\partial h_2(\boldsymbol{x},\boldsymbol{u},\dot{\boldsymbol{u}})}{\partial \boldsymbol{x}} \\ \vdots \\ \frac{\partial h_{(n-1)}\left(\boldsymbol{x},\boldsymbol{u},\dot{\boldsymbol{u}},\ldots,\boldsymbol{u}^{(n-2)}\right)}{\partial \boldsymbol{x}} \end{bmatrix} = n$$

holds for all $\boldsymbol{x} \in D_x$ *and* $\boldsymbol{u} \in C_u$.

Other than Theorem 2.10, 2.11 can be easily applied at least pointwise by computing the determinant of the observability matrix

$$\boldsymbol{Q}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right) = \frac{\partial \boldsymbol{q}\left(\boldsymbol{x}, \boldsymbol{u}, \dot{\boldsymbol{u}}, \ldots, \boldsymbol{u}^{(n-2)}\right)}{\partial \boldsymbol{x}}$$

for any $\boldsymbol{x}$ and $\boldsymbol{u}$ of interest. Note that both theorems show that the observability of nonlinear systems generally depends on the input $\boldsymbol{u}$, as was already described in Section 2.1.2. Furthermore, both theorems are only sufficient but not necessary.

Since in our case we focus on nonlinear MIMO systems, at least the above-mentioned theorem for weak observability shall be extended for multiple outputs $\boldsymbol{y}$ as it is in [28]. Before giving the formal theorem, we want to illustrate the intuition behind it.

As has been mentioned before, the observability of a system depends on whether the output and its $(n-1)$ derivatives contain enough information. (It can be shown that derivatives of order $\geq n$ do not yield any new information regarding the system's observability [29].) Multiple outputs mean that we measure additional quantities, e. g. another state variable. Therefore, amending y_1 by a new measurement y_2 so that $\boldsymbol{y}^{\mathrm{T}} = (y_1,\, y_2)$ cannot 'reduce' the observability, but only augment it. Furthermore, it seems intuitive that the observability matrix $\boldsymbol{Q}$ now contains not only y_1 and its derivatives, but also $y_2, \dot{y}_2, \ldots$ - while still only needing a rank of n. As Theorem 2.12 shows, this intuitive assumption is true.

In the following, a MIMO system of the form

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}), \tag{2.22}$$

$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}), \tag{2.23}$$

is considered. As in [28], the operator

$$L_f g_i := \frac{\partial g_i}{\partial \boldsymbol{x}} \boldsymbol{f} + \frac{\partial g_i}{\partial {}^{[n-1]}\boldsymbol{u}} \frac{\mathrm{d}^{[n-1]}\boldsymbol{u}}{\mathrm{d}t},$$
$$L_f^k g_i = L_f \left(L_f^{k-1} g_i \right), \qquad L_f^0 g_i := h_i ,$$

with

$${}^{[n-1]}\boldsymbol{u} := \left(\boldsymbol{u}^{\mathrm{T}}, \dot{\boldsymbol{u}}^{\mathrm{T}}, \ldots, \boldsymbol{u}^{(n-1)^{\mathrm{T}}} \right)^{\mathrm{T}}$$

is introduced, so that

$$\boldsymbol{z} = \begin{pmatrix} y \\ \dot{y} \\ \vdots \\ y^{(n-1)} \end{pmatrix} = \begin{pmatrix} L_f^0 \\ L_f^1 \\ \vdots \\ L_f^{n-1} \end{pmatrix} \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) = \boldsymbol{q}(\boldsymbol{x}, {}^{[n-1]}\boldsymbol{u}) . \tag{2.24}$$

Then, the observability matrix for the nonlinear MIMO system is given by

$$\boldsymbol{Q}\left(\boldsymbol{x}, {}^{[n-1]}\boldsymbol{u} \right) = \frac{\partial \boldsymbol{q}\left(\boldsymbol{x}, {}^{[n-1]}\boldsymbol{u} \right)}{\partial \boldsymbol{x}} = \begin{pmatrix} L_f^0 \\ L_f^1 \\ \vdots \\ L_f^{n-1} \end{pmatrix} \mathrm{d}\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{u}) \tag{2.25}$$

with

$$L_f \mathrm{d}g_i = \frac{\partial g_i}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}} + \boldsymbol{f}^{\mathrm{T}} \frac{\partial}{\partial \boldsymbol{x}} \left(\frac{\partial g_i}{\partial \boldsymbol{x}} \right)^{\mathrm{T}} + \frac{\mathrm{d}^{[n-1]^{\mathrm{T}}}\boldsymbol{u}}{\mathrm{d}t} \left(\frac{\partial}{\partial {}^{[n-1]}\boldsymbol{u}} \left(\frac{\partial g_i}{\partial \boldsymbol{x}} \right)^{\mathrm{T}} \right)^{\mathrm{T}} .$$

With these definitions, the following theorems can be deduced ([28]).

Theorem 2.12 *A nonlinear MIMO system described by Eqs.* (2.22) *and* (2.23) *and defined for* $\boldsymbol{x} \in D_x \subseteq \mathbb{R}^n$ *and* $\boldsymbol{u} \in C_u \subseteq \mathbb{C}^{n-1}$ *is* observable*, if the mapping* (2.24) *is uniquely invertible with respect to* $\boldsymbol{x}$.

Theorem 2.13 *A nonlinear MIMO system described by Eqs.* (2.22) *and* (2.23) *and defined for* $\boldsymbol{x} \in D_x \subseteq \mathbb{R}^n$ *and* $\boldsymbol{u} \in C_u \subseteq \mathbb{C}^{n-1}$ *is* weakly observable*, if the observability matrix from Eq.* (2.25) *has full rank n for all* $\boldsymbol{x} \in D_x$ *and* $\boldsymbol{u} \in C_u$.

Empirical Gramian Matrix

The empirical Gramian matrix was first presented by *Lall et al.* in [30] and then enhanced by *Hahn and Edgar* in [31]. Both used it as a new method to reduce nonlinear models, analogue to the nonempirical Gramian matrix for linear models. Basically, it quantifies the energy transfer from the system's states to its outputs. For a linear system with initial state $\boldsymbol{x}_0$, that would be

$$\boldsymbol{E}_0 = \int_0^\infty \boldsymbol{y}^T(\tau)\boldsymbol{y}(\tau)\mathrm{d}\tau = \boldsymbol{x}^T(0)\boldsymbol{W}_0\boldsymbol{x}(0)\ .$$

In the nonlinear case, the empirical Gramian matrix approximates this behavior. To do so, a nominal reference trajectory must be determined. Then, perturbations of the nominal states $\boldsymbol{x}_0 = \boldsymbol{x}_{\mathrm{nom}}$ must be chosen. When first introduced, the empirical observability Gramian was given as

$$\boldsymbol{W}_0 = \sum_{l=1}^{r}\sum_{m=1}^{s}\frac{1}{rsc_m^2}\int_0^\infty \boldsymbol{T}_l\boldsymbol{\Psi}^{lm}\boldsymbol{T}_l^T\mathrm{d}t \tag{2.26}$$

with

- a set $\mathscr{T}$ of r orthonormal perturbation matrices $\boldsymbol{T}_l \in \mathbb{R}^{n\times n},\ \boldsymbol{T}_i^T\boldsymbol{T}_i = \boldsymbol{I}$,
- a set $\mathscr{M}$ of s scalar constants $c_m \in \mathbb{R},\ c_i > 0$,
- the component ij of the matrix $\boldsymbol{\Psi}^{lm} \in \mathbb{R}^{n\times n}$ being

$$\Psi_{ij}^{lm}(t) = \left(\boldsymbol{y}_i^{lm} - \boldsymbol{y}_{\mathrm{nom}}\right)^T \cdot \left(\boldsymbol{y}_j^{lm} - \boldsymbol{y}_{\mathrm{nom}}\right)\ ,$$

- $\boldsymbol{y}_i^{lm}$ being the system's output corresponding to the perturbed initial condition

$$x_0^{ilm} = c_m\boldsymbol{T}_l\boldsymbol{e}_i + \boldsymbol{x}_{\mathrm{nom}} \tag{2.27}$$

- $\boldsymbol{e}_i$ being the i-th unit vector.

Please note that any inputs are neglected. Since its first introduction, the determination of $\boldsymbol{W}_0$ has been enhanced. First, *Geffen et al.* [32] suggested to normalize the system's states by adjusting Eq. (2.27) to

$$x_0^{ilm} = c_m\boldsymbol{S}^{-1}\boldsymbol{T}_l\boldsymbol{e}_i + \boldsymbol{x}_{\mathrm{nom}}\ , \tag{2.28}$$

with $\boldsymbol{S}$ being the scaling matrix. Furthermore, they remarked that instead of a set $\mathscr{T}$ with multiple matrices $\boldsymbol{T}_l \in \mathbb{R}^{n\times n}$, we could also use only one matrix $\boldsymbol{T} \in \mathbb{R}^{n\times 2^n}$ as long as it satisfies $\boldsymbol{T}^T\boldsymbol{T} = \boldsymbol{I}$. The matrix $\boldsymbol{\Psi}$ would also change its dimensions to $(2^n \times 2^n)$, but the ones of the

empirical Gramian remain the same. This way, $\boldsymbol{T}$ can be made of a full-factorial design with the elements -1 and $+1$, i. e. containing every possible combination of directions in the state space. Thus, $\boldsymbol{T}$ is given by $\boldsymbol{T} = [\boldsymbol{t}_1 \ \dots \ \boldsymbol{t}_{2^n}]$ with

$$\begin{aligned}
\boldsymbol{t}_1^T &= \frac{1}{\sqrt{2^n}}[-1 \ \dots \ -1 \ -1] \ , \\
\boldsymbol{t}_2^T &= \frac{1}{\sqrt{2^n}}[-1 \ \dots \ -1 \ +1] \ , \\
\boldsymbol{t}_3^T &= \frac{1}{\sqrt{2^n}}[-1 \ \dots \ +1 \ -1] \ , \\
\boldsymbol{t}_4^T &= \frac{1}{\sqrt{2^n}}[-1 \ \dots \ +1 \ +1] \ , \\
&\vdots \\
\boldsymbol{t}_{i_{\max}}^T &= \frac{1}{\sqrt{2^n}}[+1 \ \dots \ +1 \ +1] \ .
\end{aligned}$$

Eberle et al. ([33]) and *Glotzbach et al.* ([34]) also suggest to normalize the states with their maximum values, i. e. using $\boldsymbol{S} = \mathrm{diag}(\boldsymbol{x}_{\max})$ in Eq. (2.28) (instead of the inverse $\boldsymbol{S}^{-1}$).

In Eq. (2.26), the empirical Gramian is given with an infinite time horizon. To be computable, we need to express it as a sum of finite and discrete time points, more precisely of $k_{\max}$ simulated or measured data points with $t = \Delta T \cdot k$. Using the simplified approach described above and furthermore constraining the set $\mathscr{M}$ to a single constant c, we obtain the elements of the empirical Gramian by

$$\boldsymbol{W}_{0,ij} = (c\boldsymbol{S})^{-1} \sum_{k=1}^{k_{\max}} \boldsymbol{t}_i^T \boldsymbol{\Psi}_{ij}(k) \boldsymbol{t}_j^T \Delta T \, (c\boldsymbol{S})^{-1} \tag{2.29}$$

with

$$\boldsymbol{\Psi}_{ij}(k) = \left[\boldsymbol{y}_i(k) - \boldsymbol{y}_0(k)\right]^T \cdot \left[\boldsymbol{y}_j(k) - \boldsymbol{y}_0(k)\right]^T \ .$$

It must be mentioned that the equations above refer to the empircal *observability* Gramian, but different empirical Gramians have been introduced. In [30], an empirical controllability Gramian is used to quantify the energy transfer from the inputs to the states. *Himpe and Ohlberger* [35] summarize 4 more Gramians, which some of are mainly used for model reduction. They provide a framework for MATLAB to calculate the various matrices for a given system.

More complicated than obtaining the (observability) Gramian is its interpretation. In case of observability analysis, most of the times the observability Gramian is determined for various combinations of measured and unmeasured states. Then, it is evaluated which states need to be measured for the system to be observable, i. e. in which combination enough energy is transferred to the output. In principle, the system is observable if the Gramian has full rank.

However, since the matrix is computed numerically, it will usually be of full rank anyway. The problem is to determine 'how well of full rank' it is. Therefore, there are multiple options to interpret the matrices informative value, e. g. by its singular values. Then, the largest singular value $\overline{\lambda}(\boldsymbol{W}_0)$ refers to the state with the largest energy transfer and the smallest singular value $\underline{\lambda}(\boldsymbol{W}_0)$ to the one with the smallest energy transfer ([34]). If $\underline{\lambda}$ is too small, the system is (practically) unobservable. Unfortunately, no 'hard' lower limit can be derived but has to be defined with some kind of legitimation. *Eberle et al.* [33] e. g. assume the system to be unobservable if $\underline{\lambda}(\boldsymbol{W}_0) < 0{,}1$.

Gil [36] summarizes possible measures. In addition to looking at the individual eigenvalues or singular values, the condition number as the ratio of the biggest to the smallest singular value,

$$\mathrm{cond}(\boldsymbol{W}_0) = \frac{\overline{\lambda}(\boldsymbol{W}_0)}{\underline{\lambda}(\boldsymbol{W}_0)}, \tag{2.30}$$

is of interest. It reflects how strongly a possible (small) error in the input may change the output, whereby in our case the state spaces are the input(s). If this ratio is high, the matrix (or the system) is said to be *ill-conditioned*. If it is low, it is said to be *well-conditioned*. For easier assessment, we propose to use the inverse of the condition,

$$g(\boldsymbol{W}_0) = \mathrm{cond}^{-1}(\boldsymbol{W}_0) = \frac{\underline{\lambda}(\boldsymbol{W}_0)}{\overline{\lambda}(\boldsymbol{W}_0)}, \quad g(\boldsymbol{W}_0) \in [0,\ 1],$$

which would be close to 1 for a well-conditioned and close to 0 for an ill-conditioned observability Gramian.

The biggest problem when applying the empirical observability Gramian is the mapping of its singular values or eigenvalues to the single states. One can try to do so by analyzing the eigenvectors. However, unfortunately they are not always unambiguous enough to be allocated to a state.

Since in this work the problem of identifiability will be tackled with the one of observability at the same time, one more approach shall be summarized shortly. As is explained in Section 2.1.3 and shown in Eqs. (2.6) and (2.7), the unknown parameters of a system may be interpreted as states with no dynamics. Then, the observability Gramian $\boldsymbol{W}_{\mathrm{aug}}$ for this augmented system can be determined. *Geffen et al.* [32] suggest that if one can assume the original system to be observable, i. e. by analyzing its Gramian $\boldsymbol{W}_0$ beforehand, the part of $\boldsymbol{W}_{\mathrm{aug}}$ pertaining to the identifiability of the parameters can be extracted. With $\dim(\boldsymbol{\theta}) = q$, $\boldsymbol{W}_{\mathrm{aug}}$ can be split into

$$\boldsymbol{W}_{\mathrm{aug}}^{((n+q)\times(n+q))} = \left(\begin{array}{c|c} \boldsymbol{W}_X^{(n\times n)} & \boldsymbol{W}_{X\theta}^{(n\times q)} \\ \hline \boldsymbol{W}_{\theta X}^{(q\times n)} & \boldsymbol{W}_{\theta}^{(q\times q)} \end{array}\right)$$

and the $(q \times q)$ empirical *identifiability* Gramian then is the *Schur complement*,

$$\boldsymbol{W}_I = \boldsymbol{W}_\theta - \boldsymbol{W}_{\theta X}\boldsymbol{W}_X^{-1}\boldsymbol{W}_{X\theta}.$$

It shall be mentioned that in a similar issue, i. e. reducing the observability Gramian to only some of the states, *Glotzbach et al.* [34] simply removed the rows and columns corresponding to states which were not of interest. However, they did not justify the validity of this technique.

Local unobservability index and estimation condition number

Krener and Ide [11] use the empirical Gramian matrix in a slightly different way. Basically, they are linearizing a nonlinear system for various operating points and intend to determine the local, i. e. linear Gramian matrix for every operating point. However, they calculate the empirical Gramian matrix as an approximation.

As an important difference to most other methods presented in this thesis, they assume an autonomous nonlinear system, i. e. without any input u, given by

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t)),$$
$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}(t)).$$

That means that all system parameters $\boldsymbol{\theta}$ have to be known, too. This constraint can easily be bypassed by defining unknown parameters as state variables without dynamics as is explained in Section 2.1.3. The restriction to systems without inputs is somewhat more complicated. However, depending on the type of inputs $\boldsymbol{u}$, they might be defined as state variables, too. To do so, they need to be described by ODEs, which is possible for most of the standard control methods. More problematically, the reference input would need to be either constant or described by an ODE, too. Alternatively, the mathematical approach described below could be extended to nonlinear systems with an input vector $\boldsymbol{u}$, which is out of the scope of this work.

As mentioned before, the idea behind this approach is to assess a systems observability locally by its linearization. In [11], the term *short time locally observable* is introduced. However, there seems to be no additional information in it. A system is short time locally observable, if it is locally observable for every $t > 0$, which means that the matrix

$$\begin{bmatrix} \boldsymbol{y}(t) \\ \dot{\boldsymbol{y}}(t) \\ \vdots \\ \boldsymbol{y}^{(n-1)}(t) \end{bmatrix} = \begin{bmatrix} L_f^0 g(\boldsymbol{x}) \\ L_f^1 g(\boldsymbol{x}) \\ \vdots \\ L_f^{n-1} g(\boldsymbol{x}) \end{bmatrix}$$

has to be of rank n for every $\boldsymbol{x}$ (see [8] for validation).

The linearization of the system around trajectories $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ with a starting point $\boldsymbol{x}_0 = \boldsymbol{x}_0^*$ would be given by

$$\Delta\dot{\boldsymbol{x}} = \boldsymbol{F}(t)\Delta\boldsymbol{x}, \quad \Delta\boldsymbol{x}(t_0) = \Delta\boldsymbol{x}_0 = \boldsymbol{x}_0 - \boldsymbol{x}_0^* \tag{2.31}$$
$$\Delta\boldsymbol{y} = \boldsymbol{G}(t)\Delta\boldsymbol{x} \tag{2.32}$$

where

$$\Delta\boldsymbol{x} \approx \boldsymbol{x} - \boldsymbol{x}^*, \quad \boldsymbol{F}(t) = \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}^*(t)},$$
$$\Delta\boldsymbol{y} \approx \boldsymbol{y} - \boldsymbol{y}^*, \quad \boldsymbol{G}(t) = \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}^*(t)}.$$

Now, instead of assessing the (short time) local observability of the system only qualitatively, we can assess it quantitatively by the properties of the linearization matrix $\boldsymbol{G}(t)$.

In [11], the *local unobservability index* (LUI) is introduced as the reciprocal of the smallest singular value of the (local) $\boldsymbol{G}(t)$. The larger the LUI, the less observable is the system at this point in state space, i. e. measurement noise will have a larger impact on the observation error. Furthermore, the *local estimation condition number* (LECN) is defined as the ratio of the largest to the smallest local singular value.[9] The LECN affects the outputs sensitivity on a change (or error) in the initial condition. If it is large, a small change in one direction can swamp the effect of a change in a different direction, i. e. the observation problem is ill-conditioned near states with large LECN.

With the linearized system as in Eqs. (2.31) and (2.32) and its fundamental matrix solution $\boldsymbol{\Phi}(t)$, we could determine the local observability Gramian

$$\boldsymbol{P}(\boldsymbol{x}_0) = \int_0^{\Delta T} \boldsymbol{\Phi}^T(t)\boldsymbol{G}^T(t)\boldsymbol{G}(t)\boldsymbol{\Phi}(t)\mathrm{d}t.$$

Then, the LUI is the reciprocal of the square root of the smallest eigenvalue of $\boldsymbol{P}(\boldsymbol{x}_0)$. However, since we would need to calculate $P(x^0)$ analytically for every operating point $\boldsymbol{x}_0$, we can approximate it with the empirical local observability Gramian. Here, 'local' means that in contrast to the empirical Gramian from Eq. (2.29), we do not compute the matrix along a trajectory of data points k, but only for one time interval ΔT. Furthermore, a different perturbation

$$\boldsymbol{x}_{\pm i} = \boldsymbol{x}_0 \pm \epsilon \cdot \boldsymbol{e}_i$$

is used, with $\epsilon > 0$ and $\boldsymbol{e}_i$ being the i-th unit vector. Let $\boldsymbol{y}_{\pm i}$ be the corresponding system output. Then, every matrix entry i,j can be calculated by

$$\boldsymbol{P}(\boldsymbol{x}_0)_{i,j} = \int_0^{\Delta T} (\boldsymbol{y}_{+i}(t) - \boldsymbol{y}_{-i}(t))^T (\boldsymbol{y}_{+j}(t) - \boldsymbol{y}_{-j}(t))\mathrm{d}t.$$

Hence, as in the 'common' empirical Gramian approach described before, we only need the ability to simulate the system instead of analytically determining its linearization.

By computing the LUI and/or LECN for all points in state space of interest, we can draw a (n-dimensional) map of the systems observability, which may be a valuable extension for our system analysis. Furthermore, it is possible to split the output vector $\boldsymbol{y}$ in two vectors $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$. Then, the local observability Gramian $P(x^0)$ is the sum of the local observability Gramians $P_1(x^0)$ and $P_2(x^0)$ for $\boldsymbol{y}_1$ and $\boldsymbol{y}_2$, respectively. The changes in LUI and LECN through the additional measurements $\boldsymbol{y}_2$ are a measure of how much observability we gain by these extra observations.

[9] Please note that there is no mathematical difference to the generally known condition of a matrix, shown in Eq. (2.30).

Note, that according to [11], both state variables $\boldsymbol{x}$ and outputs $\boldsymbol{y}$ should be scaled 'properly'. If $\boldsymbol{x}$ is not scaled, the relative size of the eigenvalues of $P(x^0)$ may mean very little. If we assume additive noise, i. e. $\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{x}(t)) + \boldsymbol{\xi}(t)$, $\boldsymbol{y}$ should be scaled so that the noise covariance is the identity.

Prediction Profile Likelihood (PPL) Approach

Note: For better understanding, the reader should regard the profile likelihood approach by Raue et al. in Section 2.2.1 on page 14 first.

Kreutz et al. employed the idea and mathematical background of the profile likelihood presented in [23] and [12] to deduce confidence intervals not only for constant parameter estimates $\hat{\boldsymbol{\theta}}$, but for dynamic predictions such as state variables $\boldsymbol{x}$. This way, qualitative as well as quantitative assessments of the observability of a system based on simulated and experimental data are possible.

To do so, in contrast to other approaches such as Markov chain Monte Carlo or bootstrap based methods, no scan of the parameter space Θ is necessary. Therefore, the computational costs are much less. Instead, considering an interval of a prediction z (which could be any combination of the state variables $\boldsymbol{x}$ or just a single state variable x_i), the likelihood for every value z in this interval is computed, called the *prediction profile likelihood*, PPL(z). Due to limited space, only the mathematical statements necessary to understand the method in principle shall be explained here.

First, we need a way to compute the likelihood of measuring output data $\boldsymbol{y}$ given parameters $\boldsymbol{\theta}$. Similarly to [23] and Eq. (2.14), we use the residual sum of squares (RSS), since for Gaussian measurement noise $\boldsymbol{\varepsilon}(t) \sim N(0,\sigma^2)$ it is

$$-2\log(\mathrm{L}(\boldsymbol{y}|\boldsymbol{\theta})) = \underbrace{\sum_i \left(\frac{\boldsymbol{y}_i - \boldsymbol{M}(t_i, \boldsymbol{u}, \boldsymbol{\theta})}{\sigma} \right)^2}_{\mathrm{RSS}} + \mathrm{const},$$

with measurement data $\boldsymbol{y}_i$ at time point t_i and the model $\boldsymbol{M}$ described by Eqs. (2.12) and (2.13). That means, instead of maximizing $\mathrm{L}(\boldsymbol{y}|\boldsymbol{\theta})$, we can always just minimize the RSS.

As has been said before, z can be any combination of system variables. To formalize this idea, we define a prediction environment $D_{\mathrm{pred}} = \{t_{\mathrm{pred}}, g_{\mathrm{pred}}, \boldsymbol{u}_{\mathrm{pred}}\}$, specifying the prediction time point t_{pred} and input $\boldsymbol{u}_{\mathrm{pred}}$. g_{pred} is, in general, a function $g_{\mathrm{pred}}(\boldsymbol{x}, \boldsymbol{\theta})$ which maps the state variables to the prediction z, so that $z(t_{\mathrm{pred}}) = g_{\mathrm{pred}}(\boldsymbol{x}(t_{\mathrm{pred}}, \boldsymbol{u}_{\mathrm{pred}}), \boldsymbol{\theta})$. Formally, we write $\boldsymbol{M}(D_{\mathrm{pred}}, \boldsymbol{\theta}) = z$ if the model results in z for the parameter vector $\boldsymbol{\theta}$.

This way, the PPL can be expressed as

$$\mathrm{PPL}(z) = \min_{\boldsymbol{\theta} \in \{\boldsymbol{\theta} | \boldsymbol{M}(D_{\mathrm{pred}}, \boldsymbol{\theta}) = z\}} \mathrm{RSS}(\boldsymbol{y}_{\mathrm{data}}, \boldsymbol{\theta}),$$

and can now be computed over an interval of the prediction z, e. g. with z being a state variable x_i.

Similarly to the likelihood based confidence intervals for parameters in the profile likelihood approach, a prediction confidence interval (PCI) can be derived with the χ^2-distribution,

$$\mathrm{PCI}_\alpha(D_{\mathrm{pred}}|\boldsymbol{y}_{\mathrm{data}}) = \left\{z| -2\mathrm{PPL}(z) \leq -2\log(\mathrm{L}^*(\boldsymbol{y}_{\mathrm{data}})) + \chi^2(\alpha, df = 1)\right\},$$

where $\log(\mathrm{L}^*(\boldsymbol{y}_{\mathrm{data}}))$ is the maximum of the log-likelihood function after all parameters $\boldsymbol{\theta}$ are optimized, $\boldsymbol{\theta} = \boldsymbol{\theta}^*$. That means that if we would repeat the experiment many times, then in 95 % of the cases, the generated PCI_α would contain the true value of the prediction z.

Therefore, the computation of PPL samples values only one-dimensionally by evaluating different z's. Plus, to calculate the likelihood maximum of z, there is no unique point in parameter space Θ necessary, which would not exist for structural non-identifiabilities.

[25] also respects the possibility to validate a model by additional measurements. The true value of a model output should be in the corresponding PCI_α with a significance level α. Therefore, we ought to reject the model if a measurement is outside this interval. However, since every measurement is noisy, we should respect the measurement noise with its variance SD^2. Note, that we already respected measurement noise $\boldsymbol{\varepsilon}(t) \sim N(0,\sigma^2)$ when calculating the RSS, but consider the possibility of different measurement noise for the prediction, such that $z_{\mathrm{vali}} \sim \mu + \boldsymbol{\varepsilon}(t)_z = M(D_{\mathrm{vali}}, \boldsymbol{\theta}_{\mathrm{true}}) + N(0, \mathrm{SD}^2)$.

Since a validation by additional (real) measurements is not our purpose, only necessary equations shall be given. The calculation of the log-likelihood respects the RSS of both the before used data $\boldsymbol{y}_{\mathrm{data}}$ and the new validation data point.

$$\log(\mathrm{L}(z, \boldsymbol{y}|\boldsymbol{\theta})) = \underbrace{\sum_i \left(\frac{\boldsymbol{y}_i - \boldsymbol{M}(t_i, \boldsymbol{u}, \boldsymbol{\theta})}{\sigma^2}\right)^2}_{\text{RSS of data } y} + \underbrace{\left(\frac{z - \boldsymbol{M}(D_{\mathrm{vali}}, \boldsymbol{\theta})}{\mathrm{SD}}\right)^2}_{\text{RSS of validation data point } z} \tag{2.33}$$

Let again be $\boldsymbol{\theta}^*$ the maximizing argument for Eq. (2.33). Then, the *validation profile log-likelihood* (VPL) is defined by

$$\mathrm{VPL}^{\mathrm{SD}}(z|\boldsymbol{y}) = \log(\mathrm{L}(z,\boldsymbol{y}|\boldsymbol{\theta}^*)),$$

and the validation confidence interval (VCI) is given by

$$\mathrm{VCI}^{\mathrm{SD}}_\alpha(D_{\mathrm{vali}}|\boldsymbol{y}) = \left\{z| -2\mathrm{VPL}^{\mathrm{SD}}(z|\boldsymbol{y}) \leq -2\log(\mathrm{L}(z,\boldsymbol{y}|\boldsymbol{\theta}^*)) + \chi^2(\alpha, df = 1)\right\}.$$

Figure 2.3 shows the computed PCIs and VCIs for a model used in [25]. As can be seen, the PCIs for some state variables are significantly bigger than for others. Especially the PCI for Mek*, being ≈ 3 times the size of Mek* itself, indicates how poorly observable it is. Some state variable as Raf, Raf*, Mek and Erk seem to be well observable. Others, as Erk*, Erk** and Mek** fluctuate in their observability, i. e. their PCIs mostly grow over time.

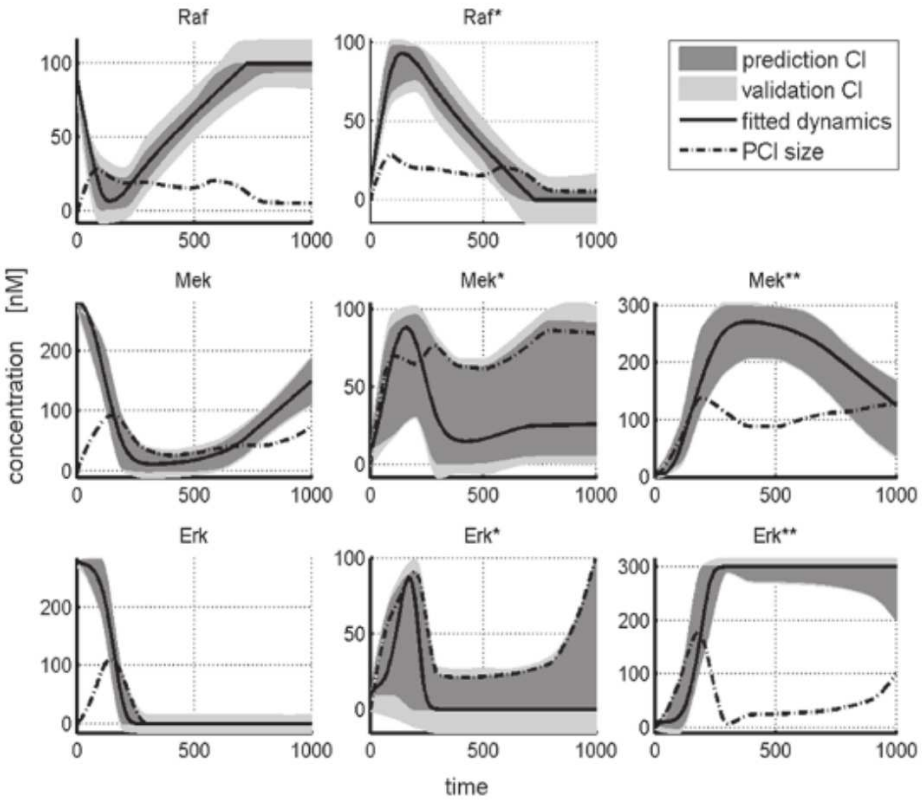

Figure 2.3.: Examplary PCIs and VCIs. Source: [25]

Again, an implementation of the prediction profile likelihood by its authors, along with several other methods as the profile likelihood approach, is available for MATLAB as part of the *Data2Dynamics* framework [37]. Therefore, we do not need to be implement it ourselves. However, using the framework, only this specific implementation can be evaluated, since it cannot be guaranteed that the implementation does not effect the results. The application of the PPL to our models can be found in Section 4.4.

Markov chain Monte Carlo (MCMC) Methods

MCMC methods are part of Bayesian statistics, which is mostly based on Bayes' theorem and represents an own genre in statistics which engineers usually are not too experienced in. Therefore, we rather try to give an intuitive introduction to the MCMC approach than exploring mathematical details. Before introducing an application of MCMC methods to an identifiability and observability problem of a non-linear system like ours [38], the principles are explained in a general manner.

Bayes' theorem says that the probability of an event A given the outcome of another event B, which might affect the likelihood of A, can be expressed by the inverse case and the single probabilities of A and B,

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} .$$

In our case, we (first) want to identify the parameters $\boldsymbol{\theta}$ of a system given the experimental data $\boldsymbol{y}$. For simplicity and without restriction of generality, we assume to have only one parameter θ and a data set y. Then, Bayes' theorem gives

$$P(\theta|y) = \frac{P(y|\theta) \cdot P(\theta)}{P(y)} \,. \tag{2.34}$$

Albeit being a scalar parameter, we can interpret θ as a random variable. Then, we define a prior density function $P(\theta)$ which reflects whatever knowledge we have about θ. This might be a flat distribution with bounds if no further knowledge is available. Then, our goal is to update the prior density function $P(\theta)$ to the posterior $P(\theta|y)$, i. e. after the data y has been measured.

With this assumed prior $P(\theta)$, we could compute the probability $P(y|\theta)$ to measure y given $P(\theta)$ in Eq. (2.34). However, in general there is no way to analytically compute the denominator

$$P(y) = \int_{\Theta} P(y,\theta) \mathrm{d}\theta \,.$$

MCMC-methods propose a tool to avoid the computation of the denominator. Instead, we focus on receiving a posterior probability density $P(\theta|y)$ by sampling the parameter space Θ of θ in a more or less smart way, depending on the chosen method.

To do so, we start at a randomly chosen initial value θ. Furthermore, we once choose a *proposal distribution*, which determines how we jump in the parameter space. Commonly, a normal distribution is chosen with mean zero and a step width σ^2_{step} as its variance. Having a proposed value θ_{prop} (somewhere around the 'old' value θ), we have to evaluate how 'good' it is, i. e. how high the probability is that $\theta_{\text{prop}} = \theta_{\text{true}}$, which means computing the posterior $P(\theta|y)$ - which we cannot, due to the denominator $P(y)$.

However, we can compare the posterior probability density of the old value θ and the new proposed value θ_{prop}, since the denominator cancels out,

$$\frac{P(\theta_{\text{prop}}|y)}{P(\theta|y)} = \frac{\frac{P(y|\theta_{\text{prop}}) \cdot P(\theta_{\text{prop}})}{P(y)}}{\frac{P(y|\theta) \cdot P(\theta)}{P(y)}} = \frac{P(y|\theta_{\text{prop}}) \cdot P(\theta_{\text{prop}})}{P(y|\theta) \cdot P(\theta)}. \tag{2.35}$$

Since we do have a model for y based on θ, we can calculate $P(y|\theta)$ and we also do have a probability density $P(\theta)$ that we assumed before. Therefore, we are able to compute the relation of the two probability densities in Eq. (2.35).

Now, if the posterior probability density for θ_{prop} is higher, we should want to accept the new step, otherwise reject it. However, using a hard rule like this would result in a hill-climbing algorithm. That means, if there were no local minimums, we would eventually get to θ_{true}. Otherwise, we might get stuck in a local minimum. However, our goal is to get the posterior $P(\theta|y)$. Therefore, we accept every step which gets us to a higher probability, i. e. if $\frac{P(\theta_{\text{prop}}|y)}{P(\theta|y)} \geq 1$, but do sometimes also accept a step that gets us to a lower probability. To do so, we compare the

resulting relation to a random number γ between 0 and 1. That means, we accept the new step if

$$\min\left(\frac{P(\theta_{\text{prop}}|y)}{P(\theta|y)}, 1\right) > \gamma \ .$$

Doing so, it can be mathematically proven that the samples which are generated this way, i. e. all the θ's we accept over time, follow the same probability density as the posterior $P(\theta|y)$ of our model which we are looking for. However, it is important to notice that this behavior arises only after a burn-in period which has to be 'cut off'. That is due to a (probably) wrong starting point θ_{init}. If we started at the real posterior of θ, there would be no burn-in period.

The jumping to the next point in parameter space based only on the position before is the 'Markov chain'-part in MCMC, since the likelihood of θ_{prop} does not depend on any stage before the current one. The 'Monte Carlo'-part is the (more or less) randomly scanning of the whole parameter space. The above explained sampling behavior refers to the *Metropolis-Hastings-Algorithm*, which is the simplest MCMC algorithm. Obviously, there are more complicated algorithms which aim on improving the efficiency of the algorithm.

Now, we consider a nonlinear system such as given in Eqs. (2.12) and (2.13). For clearance, we refer to the parameter vector $\boldsymbol{\theta}$ and measured data $\boldsymbol{y}_{\text{data}}$ again. The posterior $P(\boldsymbol{\theta}|\boldsymbol{y}_{\text{data}})$ has been calculated as described above. Then, a *posterior predictive distribution* (PPD), i. e. the likelihood to obtain data $\boldsymbol{y}$ in a consecutive experiment given the yet measured data $\boldsymbol{y}_{\text{data}}$ from an experiment before, can be introduced. In [38], it is given by

$$P(\boldsymbol{y}|\boldsymbol{y}_{\text{data}}, \boldsymbol{u}(t)) = \int P(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{u}(t))\, P(\boldsymbol{\theta}|\boldsymbol{y}_{\text{data}})\, \mathrm{d}\boldsymbol{\theta} \ . \tag{2.36}$$

$P(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{u}(t))$ can be obtained by simulation of the system with $\boldsymbol{\theta}$ and $\boldsymbol{u}(t)$, whereas the additive measurement noise must be respected. That means we can now give a prediction of the output $\boldsymbol{y}$ for an arbitrary input $\boldsymbol{u}(t)$ based on measured data in the past.

However, since we obtain $P(\boldsymbol{y}|\boldsymbol{\theta}, \boldsymbol{u}(t))$ by simulation, we can compute distributions for any prediction, e. g. for the state variables $\boldsymbol{x}$ by substituting $\boldsymbol{y}$ with $\boldsymbol{x}$ in Eq. (2.36). This way, their distribution (and especially their variance) are a quantification of their observability. Furthermore, by computing the PPDs for different constellations of measurements, the efficacy of possible new measurements can be assessed, as is illustrated in Figure 2.4 .

In the the top left and low right quadrants, the PPDs for two quantities A and B are shown, respectively. The dark distributions represent the initial constellation where none of them are measured. The light distributions represent the case when A is measured.[10] As one can see, A still has a distribution due to measurement noise. On the other hand, the distribution of B is not only deferred, but also narrower. The upper right quadrant shows the combined PPD for A and B for the initial case, i. e. without the measurement of A. The white dots show evaluated samples from the Markov Chain. By plotting the PPDs for the relevant quantities, the efficacy of

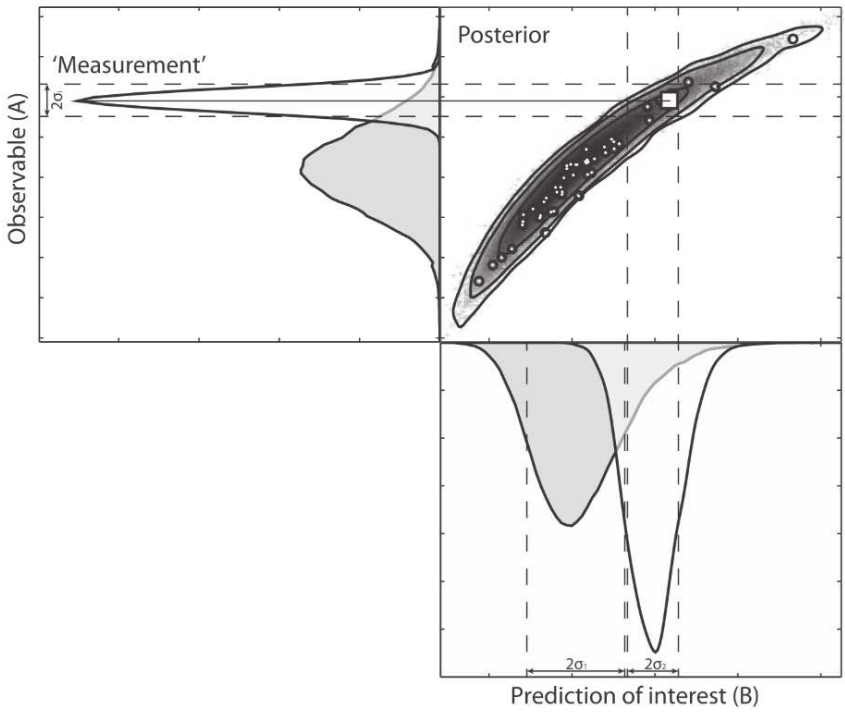

Figure 2.4.: Comparison of PPDs for two different measurement constellations. Source: [38]

new measurements can be shown visually. In order to facilitate a quantitative assessment, one can consider the expected reduction in variance. First, we calculate the expectation value of an arbitrary quantity of interest z, e. g. a state variable x, which is given by

$$E(z) = \frac{1}{N}\sum_{r=1}^{N}\sum_{i=1}^{N}\frac{G(t,\boldsymbol{u}(t),\boldsymbol{\theta}_i,\boldsymbol{\theta}_r)}{\sum_{k=1}^{N}G(t,\boldsymbol{u}(t),\boldsymbol{\theta}_k,\boldsymbol{\theta}_r)}z(t,\boldsymbol{u}(t),\boldsymbol{\theta}_i)\,,$$

$$G(t,\boldsymbol{u}(t),\boldsymbol{\theta}_i,\boldsymbol{\theta}_r) = e^{-\frac{(\boldsymbol{y}(t,\boldsymbol{u}(t),\boldsymbol{\theta}_i)-\boldsymbol{y}(t,\boldsymbol{u}(t),\boldsymbol{\theta}_r))^2}{2\sigma^2}}\,,$$

where $\boldsymbol{\theta}_j$ is the j-th parameter vector in the Markov chain, N the total number of MCMC samples and G refers to a Gaussian error model with SD σ.

Now, being able to calculate $E(z)$, we can calculate the variance of z by

$$\mathrm{Var}(z) = E(z^2) - (E(z))^2\,.$$

The *sampled variance reduction* (SVR) is then given by

$$\mathrm{Var}R = 1 - E\left(\frac{\sigma^2_{\mathrm{new}}}{\sigma^2_{\mathrm{old}}}\right).$$

[10] Since the outcome of the new measurement, i. e. including A, is not known before the actual experiment, a different way to estimate the impact of the new measurement on the PPD is needed. One solution would be to substitute every possible value of A as the 'true' output, compute the MCMC for every value again and average the results. However, a single MCMC run is already computationally demanding. For an alternative approach, the reader is referred to [38].

An implementation of the MCMC method for MATLAB used in [38] is available in the *Data2Dynamics* package from [37]. However, in practice, real advantages of MCMC come with prior knowledge of the parameter and especially the error distributions. Unfortunately, in our case the greater part of errors comes from modeling errors. Therefore, the MCMC approach is not applied in this work.

3 Physical Modeling and Methodology

This chapter provides all necessary information about the modeling and methodology used in this work. First, two different non-linear wind turbine models are presented. Secondly, a short introduction into the used MATLAB framework D2D is given. Thirdly, our models and methods implementation in D2D is explained. Lastly, an overview of different scenario variations that are and can be analyzed is provided.

3.1 Wind Turbine Modeling

In this section, two nonlinear wind turbine models of different complexity are presented. First, the basic model with three states is presented. Then, the standard wind turbine model with eight states is specified. The full nomenclature for both models can be found in the *Symbols and Abbreviations* chapter on page vii.

3.1.1 Basic Wind Turbine Model

The first model to analyze is a simplified wind turbine model with three dynamic states described by the following two ODEs

$$\ddot{\varphi}_{\mathrm{g}} = \frac{\varrho}{2}\frac{\pi R^3}{\Theta} C_{\mathrm{M}}(\lambda)(v_{\mathrm{w}} - \dot{x}_{\mathrm{T}})^2 - \frac{i_{\mathrm{gb}}}{\Theta} M_g \,, \tag{3.1}$$

$$\ddot{x}_{\mathrm{T}} = \frac{\varrho}{2}\frac{\pi R^2}{m_{\mathrm{T}}} C_{\mathrm{T}}(\lambda)(v_{\mathrm{w}} - \dot{x}_{\mathrm{T}})^2 - 2\zeta_{\mathrm{T}x}\omega_0 \dot{x}_{\mathrm{T}} - \omega_0^2 x_{\mathrm{T}} \,, \tag{3.2}$$

with

$$\lambda = \frac{\dot{\varphi}_{\mathrm{g}} R}{v_{\mathrm{w}} - \dot{x}_{\mathrm{T}}} \,. \tag{3.3}$$

Therein, the state space vector, the outputs and inputs are defined as

$$\boldsymbol{x} = \begin{bmatrix} \dot{\varphi}_{\mathrm{g}} & \dot{x}_{\mathrm{T}} & x_{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^3,$$
$$\boldsymbol{y} = \begin{bmatrix} n_{\mathrm{g}} & \ddot{x}_{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^2,$$
$$\boldsymbol{u} = \begin{bmatrix} M_g & v_{\mathrm{w}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^2,$$

whereby the generator speed is given by $n_g = \frac{60}{2\pi} i_{\mathrm{gb}} \dot{\varphi}_{\mathrm{g}}$, i. e. it is directly proportional to the angular speed (and state) $\dot{\varphi}_{\mathrm{g}}$. The vector $\boldsymbol{\theta}$ of uncertain parameters will be varied during the

analysis. The wind speed v_w may either be interpreted as an input or as an unknown (constant) parameter.

To avoid the necessity of characteristic diagrams, the model is only defined in a partial load regime of $v_\mathrm{w} \in [0, 11.4\frac{\mathrm{m}}{\mathrm{s}}]$ and $\lambda \in [4, 10]$. Then, the aerodynamic characteristics can be approximated by

$$C_\mathrm{M}(\lambda) = c_{\mathrm{m},2}\lambda^2 + c_{\mathrm{m},1}\lambda + c_{\mathrm{m},0}\,, \tag{3.4}$$

$$C_\mathrm{T}(\lambda) = c_{\mathrm{t},2}\lambda^2 + c_{\mathrm{t},1}\lambda + c_{\mathrm{t},0}\,, \tag{3.5}$$

with

$$c_{\mathrm{m},2} = -4{\cdot}10^{-4},\ c_{\mathrm{m},1} = -1.7{\cdot}10^{-3},\ c_{\mathrm{m},0} = 0.1\,,$$
$$c_{\mathrm{t},2} = -7.7{\cdot}10^{-3},\ c_{\mathrm{t},1} = 0.18,\ c_{\mathrm{t},0} = -0.16\,.$$

3.1.2 Standard Wind Turbine Model

The second model we want to analyze is the standard version with 8 states [39],

$$\boldsymbol{x} = \begin{bmatrix} \dot{x}_\mathrm{T} & \dot{y}_\mathrm{T} & \dot{\varphi}_\mathrm{g} & \Delta\dot{\varphi} & x_\mathrm{T} & y_\mathrm{T} & \varphi_\mathrm{g} & \Delta\varphi \end{bmatrix}^\mathrm{T} \in \mathbb{R}^8\,.$$

Additionally to the generator torque and the wind speed, the three blade pitches are part of the input vector,

$$\boldsymbol{u} = \begin{bmatrix} M_\mathrm{g} & v_\mathrm{w} & \beta_1 & \beta_2 & \beta_3 \end{bmatrix}^\mathrm{T} \in \mathbb{R}^5\,.$$

The measured outputs are given by

$$\boldsymbol{y} = \begin{bmatrix} \ddot{x}_\mathrm{T} & \ddot{y}_\mathrm{T} & n_\mathrm{g} & \varphi \end{bmatrix}^\mathrm{T} \in \mathbb{R}^4\,.$$

Since the first four states are the derivatives of the last four, we only need four more ODEs to describe the system,

$$m_\mathrm{T}\ddot{x}_\mathrm{T} + b_{\mathrm{T}x}\dot{x}_\mathrm{T} + k_{\mathrm{T}x}x_\mathrm{T} = \frac{\varrho}{2}\frac{\pi R^2}{3}\sum_{b=1}^{3}\Big(1 + \zeta r_\mathrm{n}\cos\psi_b\Big)C_\mathrm{T}(\lambda_b,\beta_b)v_b^2\,, \tag{3.6}$$

$$-m_\mathrm{T}\ddot{y}_\mathrm{T} - b_{\mathrm{T}y}\dot{y}_\mathrm{T} - k_{\mathrm{T}y}y_\mathrm{T} = \frac{\varrho}{2}\frac{\pi R^3}{3\,r_\mathrm{t}}\sum_{b=1}^{3}\cos\psi_b\,C_\mathrm{M}(\lambda_b,\beta_b)v_b^2 + \zeta\, i_\mathrm{gb}M_\mathrm{g}\,, \tag{3.7}$$

$$\Theta_\mathrm{r}(\ddot{\varphi}_\mathrm{g} + \Delta\ddot{\varphi}) + \Theta_\mathrm{g}\ddot{\varphi}_\mathrm{g} = \frac{\varrho}{2}\frac{\pi R^3}{3}\sum_{b=1}^{3}C_\mathrm{M}(\lambda_b,\beta_b)v_b^2 - i_\mathrm{gb}M_\mathrm{g}\,, \tag{3.8}$$

$$\Theta_\mathrm{r}(\ddot{\varphi}_\mathrm{g} + \Delta\ddot{\varphi}) + b_\varphi\Delta\dot{\varphi} + k_\varphi\Delta\varphi = \frac{\varrho}{2}\frac{\pi R^3}{3}\sum_{b=1}^{3}C_\mathrm{M}(\lambda_b,\beta_b)v_b^2\,. \tag{3.9}$$

Therein, the blade azimuth angle ψ_b, the blade effective wind speed v_b and the tip speed ratio λ_b are given by

$$\begin{aligned}
\psi_b &= \varphi_g + \Delta\varphi + 2\pi/3\,(b-1)\,, \\
v_b &= \Big(1 + H^{-1} r_{\text{B}} \cos\psi_b\Big)^{\kappa} v_{\text{w}} - \Big(1 + \zeta\, r_{\text{B}} \cos\psi_b\Big)\dot{x}_{\text{T}} \quad \text{and} \\
\lambda_b &= \Big(\dot{\varphi}_{\text{g}} + \Delta\dot{\varphi}\Big) R\, v_b^{-1} = \Omega R\, v_b^{-1}\,.
\end{aligned}$$

Assuming $k_{\text{T}x} \approx k_{\text{T}y}$, we can replace the fore-aft and side-side stiffness and damping coefficients with

$$\omega_0^2 = \frac{k_{\text{T}x}}{m_{\text{T}}} = \frac{k_{\text{T}y}}{m_{\text{T}}}, \quad \zeta_{\text{T}x} = \frac{1}{2}\frac{b_{\text{T}x}}{m_{\text{T}}}\frac{1}{\omega_0}, \quad \text{and} \quad \zeta_{\text{T}y} = \frac{1}{2}\frac{b_{\text{T}y}}{m_{\text{T}}}\frac{1}{\omega_0}\,,$$

which is necessary to identify the eigenfrequency ω_0 as a single parameter later on. By using this nomenclature and converting Eqs. (3.6)-(3.9), we get

$$\begin{aligned}
\ddot{x}_{\text{T}} &= \frac{\varrho}{6}\frac{\pi R^2}{m_{\text{T}}}\sum_{b=1}^{3}\Big(1 + \zeta\, r_{\text{n}}(v_{\text{w}})\cos\psi_b\Big) C_{\text{T}}\big(\lambda_b,\beta_b\big) v_b^2 \; - 2\zeta_{\text{T}x}\omega_0\dot{x}_{\text{T}} - \omega_0^2 x_{\text{T}}\,, \\
\ddot{y}_{\text{T}} &= -\frac{\varrho}{6}\frac{\pi R^3}{m_{\text{T}} r_{\text{t}}}\sum_{b=1}^{3}\cos\psi_b\, C_{\text{M}}\big(\lambda_b,\beta_b\big) v_b^2 \; - \frac{\zeta\, i_{\text{gb}}}{m_{\text{T}}} M_{\text{g}} - 2\zeta_{\text{T}y}\omega_0\dot{y}_{\text{T}} - \omega_0^2 y_{\text{T}}\,, \\
\ddot{\varphi}_{\text{g}} &= \frac{1}{\Theta_{\text{g}}}\Big(b_\varphi\Delta\dot{\varphi} + k_\varphi\Delta\varphi - i_{\text{gb}} M_{\text{g}}\Big)\,, \\
\Delta\ddot{\varphi} &= \frac{\varrho}{2}\frac{\pi R^3}{3\Theta_{\text{r}}}\sum_{b=1}^{3} C_{\text{M}}\big(\lambda_b,\beta_b\big) v_b^2 + \frac{i_{\text{gb}}}{\Theta_{\text{g}}} M_{\text{g}} - \frac{\Theta_{\text{r}}+\Theta_{\text{g}}}{\Theta_{\text{r}}\Theta_{\text{g}}}\Big(b_\varphi\Delta\dot{\varphi} + k_\varphi\Delta\varphi\Big)\,.
\end{aligned}$$

Note that in the partial load regime, i. e. if $v_{\text{w}} \in [0, 11.4\frac{\text{m}}{\text{s}}]$, the aerodynamic characteristics $C_{\text{M}}\big(\lambda_b,\beta_b\big)$ and $C_{\text{T}}\big(\lambda_b,\beta_b\big)$ can be approximated by the polynomials given in Eqs. (3.4) and (3.5).

3.2 Data2Dynamics Principles

Data2Dynamics (D2D) is an open-source framework for MATLAB, mainly developed by researchers from the University of Freiburg [37] and is currently hosted on GitHub.[1] It can be used to simulate systems described either by ODEs or a reaction network, as is often the case in biological systems. To improve computational efficiency, the model equations are compiled to *MEX* files and the actual calculations are done in *C*, partly parallelized.[2] Furthermore, many methods are already implemented, e. g. the profile likelihood (PL) and prediction profile likelihood (PPL) approach. To the author's knowledge, it has mainly been used for identification purposes of biological systems with many uncertainties, i. e. unknown parameters and noisy measurements, and few known data. Therefore, some enhancements were necessary during

[1] `https://github.com/Data2Dynamics`

the progress of this work and we would like to thank *Helge Hass* and *Joep Vanlier* from the Institute of Physics, University of Freiburg equally for their implementation.

Introductionary material can be found in the supplementary data[3] to [37] and in the GitHub wiki. Since its documentation is sparse and the use of a mechanical model uncommon, the following sections summarize the necessary information to use both the framework and the model implementation itself that has been a result of this work.

The framework itself comes as a composition of MATLAB and *C* files that need to be added to your MATLAB path. Furthermore, your project folder needs to contain at least one 'Models' and one 'Data' folder. For both model and data, a text file with the ending *.def* is necessary. They need to follow a predefined structure shown in Table 3.1, whereas the *data.def* file may lack some sections.

Table 3.1.: Necessary sections of the *model.def* file

```
DESCRIPTION
...
PREDICTOR
...
COMPARTMENTS
...
STATES
...
INPUTS
...
ODES
...
DERIVED
...
OBSERVABLES
...
ERRORS
...
CONDITIONS
...
```

[2] To use the parallel computation on a Windows machine, two *.dll* files need to be placed in the *C:\Windows* folder. For more details, see `https://github.com/Data2Dynamics/d2d/wiki/Installation`

[3] `https://academic.oup.com/bioinformatics/article/31/21/3558/195191/Data2Dynamics-a-modeling-environment-tailored-to`

In the following, we will explain in summary fashion every section with respect to our use case. Since D2D is developed mainly for biological systems, some specifications are not relevant for our purposes.

- DESCRIPTION
  This section only contains meta information given as quoted lines of text, e. g.
  ```
  "3 state space model"
  "Version:  1.0"
  ```
- PREDICTOR
  Here, we can define the time unit and horizon for which the system should be simulated, e. g.
  ```
  t   T   "s"   "time"   0   10
  ```
- COMPARTMENTS
  Compartments are only relevant for biological systems. Therefore, this section is to be left empty.
- STATES
  Here, every system state is listed in an own line with 7 more arguments. The 2nd-4th argument are the unit type, the unit itself and a plain text used for plots, i. e. they are arbitrary for us. The 5th argument defines the compartment. Since there is no compartment in our model, it should be defined as `default`. The 6th is a flag if the state should be showed in plots (0=no, 1=yes). The 7th argument is a clear text label for plotting and therefore arbitrary again. However, the 8th argument is a flag whether the state is strictly positive (0=no, 1=yes) and therefore very important. For example, the state variable $\dot{\varphi}_g$ would be defined as
  ```
  QD_GeAz   C   "rad/s"   "angular speed"   default   1   "QD_GeAZ"   0
  ```
- INPUTS
  Usually, in this section the inputs are defined in form of mathematical expressions. Therefore, the name is followed by arguments 2-4 as in the STATES section before and ends with the mathematical expression enclosed by " ". However, in our case the inputs are usually given as data points. Therefore, we express them as a large spline that connects every data point by using the function
  `inputspline(t, n, [`$t_0, t_1, \ldots, t_n$`], [`$u(t_1), u(t_2), \ldots, u(t_n)$`])`.
  The input M_g could look like
  ```
  GenTq C "Nm" "generator torque" "inputspline(t, 3, [0, 0.01, 0.02],
  [3.14, 3.27, 3.19])"
  ```
- ODES
  For biological systems, this section would alternatively be called REACTIONS. In our case, the differential equations are given in the same order as the states, again enclosed by " ". For the derivative $\ddot{\varphi}_g$ of $\dot{\varphi}_g$, it would look like

```
"AirDensity/2*PI*BldTpRd^3/(HubInertia+BldInertia1+ BldInertia2+
BldInertia3 +GenInertia*GbRatio^2) ... "
```

- DERIVED

  Here, auxiliary variables derived from input and state variables can be defined. In our case, this section can be left empty.

- OBSERVABLES

  In this section, the observed, i. e. measured quantities are defined. The name is followed by the arguments 2-4 as in the STATES section. Argument 5 is a flag whether the maximum value given in the data sheet, which will be explained later on, should be rescaled to 1 (0=no, 1=yes). The 6th argument is a flag indicating whether the observable is given on a log10-scale (0=no, 1=yes). In our case we always define both flags as 0. Lastly, the mathematical expression of the observable is given, again enclosed by " ", e. g. only the name of a state, a combination of states or a complete ODE in case the derivative of a state is measured. For the 3 state space model, next to the rpm n_g, the towers fore-aft acceleration $\ddot{x}_{\mathrm{T}}$ is measured, i. e. the complete ODE is to be repeated:

  ```
  y1_NcIMUTAxs   C   "m/s^2"   "fore-aft acc."   0   0   "AirDensity/2*
  PI* BldTpRd^3/ (HubInertia +BldInertia1+ BldInertia2+ BldInertia3
  +GenInertia*GbRatio^2) ... "
  ```

- ERRORS

  For the PL and PPL approach, a gaussian distributed measurement noise needs to be assumed. In this section, the corresponding standard deviation for every observable has to be expressed as a variable or a parametrized function. In our case, we only define a variable as an absolute error, e. g.

  ```
  y1_NcIMUTAxs   "NcIMUTAxs_abs"
  ```

- CONDITIONS

  Here, model parameters can be defined as a mathematical expression, which can also be only a numerical value. That means every parameter or term in one of the mathematical expressions above that is not a state or input will be treated as a free parameter, unless otherwise defined in this section. In every line, a parameter name has to be typed followed by the corresponding mathematical expression (or numerical value), again enclosed by " ", e. g.

  ```
  BldTpRd        "63"
  GbRatio        "97"
  init_QD_GeAz   "0"
  ```

  Please note that if a states initial value should not be estimated as every other free parameter, it has to be defined here, too. Its term is always `init_` followed by the states name.

Furthermore, another section called PARAMETERS could be defined. However, its intention is only to save final parameter values for documentation purpose and is therefore unnecessary. A full *model.def* file for the 3 state space model can be found in Appendix A.

In addition to the *model.def* file, a *data.def* file has to be set up. However, its purpose is only to define different experiment settings and can nearly be left empty in our case. Only the PREDICTOR section with the definition of the simulation time has to be the same as in the *model.def* file, as is shown in Table 3.2.

Table 3.2.: *data.def* file. Only the PREDICTOR section needs to be defined and is the same as in the *model.def* file.

```
DESCRIPTION
"arbitrary"
PREDICTOR
  t   T   "s"   "time"   0   10
INPUTS

OBSERVABLES

ERRORS

CONDITIONS

```

In addition, measurement data of the observables has to be stored in either a *.csv* or a *.xls* file in the 'Data' folder. Since we generate all files automatically in MATLAB, the easier to handle *.csv* format is chosen. It is mandatory that it has the same name as the *data.def* file.

Table 3.3 shows the necessary structure. In the first line, 'Time' and the observables names are typed. In the following lines, the corresponding values are given. Theoretically, measurement errors in form of a standard deviation could be given by adding extra columns with the observables names followed by '_std'. Please note that in this case, the *model.def* file may not contain any additional conditions in the ERRORS section.

To actually run the model, all mentioned files need to be loaded and compiled. This is done in a regular MATLAB-script with a few commands. Assuming a model file *myModel.def* and data files *myModel_data.def* and *myModel_data.csv*, the necessary code is given in Listing 3.1.

Listing 3.1: Initializing, loading and compiling a model in D2D

```
1 arInit;
  arLoadModel('myModel');
  arLoadData('myModel_data', 'myModel', 'csv');
```

Table 3.3.: *data.csv* file.

Time,	y1_GenSpeed,	y2_NcIMUTAxs,
0,	944.95,	0.95781
0.01,	944.98,	0.95721
0.02,	945,	0.95618
0.03,	945.03,	0.95474
0.04,	945.05,	0.95288
0.05,	945.07,	0.95061
0.06,	945.09,	0.94553
0.07,	945.11,	0.93949
0.08,	945.12,	0.93266
0.09,	945.13,	0.92518

```
arCompileAll(true);
```

The compilation time heavily depends on the models complexity and the amount of data. It can vary from 40 s up to several minutes. Once the model is compiled, the actual methods can be applied. Since at least one free parameter is necessary, usually parameter estimation is done first. The parameter estimation can either be done with only one iteration by `arFit` or with n random starting values and iterations by `arFitLHS(n)`. Further optimization details will be discussed in Section 3.3.2.

It is important to mention that after initializing and compiling the model, all data is stored and edited in a global variable `ar`, i. e. every setting as well as every result are saved in this structure. Unfortunately, this way `ar` can become confusing. An overview of the most important fields is given in the online wiki.[4] After fitting the free parameters, both PLE[5] and PPL can be performed separately from each other.

3.3 Model and Data Set Implementation

After the principles of the D2D framework have been explored, a detailed look at the realized implementation of the physical models is necessary. First, the implementation of the inputs, ODEs and observation equations is explored, i. e. everything that is stored in the *model.def*, *data.def* and *data.csv* files. Then, the chosen settings for parameter optimization are discussed. Lastly, the application of both the profile likelihood and prediction profile likelihood methods are described.

[4] `https://github.com/Data2Dynamics/d2d/wiki/What%20are%20the%20most%20important%20fields%20of%20the%20variable%20ar%3F`

[5] The MATLAB function computing the profile likelihood is called `ple`, as in profile likelihood exploit. Thus, in the following we sometimes refer to PLE when the profile likelihood method is meant.

3.3.1 Inputs, ODEs and Observations

As has been mentioned before, the D2D framework has not been developed for mechanical but mainly for biological models. Therefore, it is not designated to process new input data for every time point. If you look through the many examples,[6] different input scenarios are nearly always defined in the *data.csv* file. However, if you do so, every different combination of measurement and input data is interpreted as a different experiment setting and therefore simulated independent from the remaining data. Hence, we need to define the input data directly in the *model.def* file as a mathematical function.

A possible workaround that has been used is to concatenate the data points by splines. However, the maximum number of points a spline could contain was 10, using the built-in `monospline10` function. Furthermore, the splines started before the first and did not end after the last point and needed to be windowed with step functions. This approach does work and is still implemented as a possible setting, but is computationally very demanding and has been replaced by the new `inputspline`-function that was implemented in D2D during this work.[7] Now, all `n` data points can be combined by one single spline, i. e.

`intputspline(t, n, [`$t_0, t_1, \ldots, t_n$`], [`$u(t_1), u(t_2), \ldots, u(t_n)$`])`,

assigning the input values in the second interval to the time points in the first interval. The maximum value for `n` is set to 10,000 by default and can be changed in the framework.[8] Furthermore, by setting a flag in the matlab script

```
ar.config.turboSplines = 1;
```

the framework uses caching for the splines and is up to three times faster at runtime and during fitting. To generate the necessary strings for the *model.def* file automatically, we implemented a function `writeSuperSplines(t_data, u_data)` that takes a time and a data array as inputs and constructs an appropriate string.

For both models the generator torque is given as a known input for every time point. For the standard model with eight states, the three blade pitches are defined as known inputs, too. However, in the partial load regime $v_w \in [0, 11.4\frac{m}{s}]$, all blade pitches are equal to zero, $\beta_1 = \beta_2 = \beta_3 = 0$. Therefore, the aerodynamic characteristic can be described by a polynomial approximation as in Eqs. (3.4), (3.5) and the blade pitches are unused. The wind velocity v_w is either defined as a known input or estimated as an unknown (constant) parameter.

The ODEs and observables could be written into the *model.def* file directly. Again, for automation purposes, a function[9] links every dynamic variable to an equation defined in it. That means that the necessary equations need to be typed manually only once. Then, our MATLAB script

[6] The examples can be found in the framework structure, *arFramework3\Examples*

[7] Again, credits go to Joep Vanlier.

[8] To do so, we have to change the constant `MAX_LONG_SPLINE` in `arFramework3\Ccode\monotone.c`

[9] `GenerateDefD2D_3Z_Modell` and `GenerateDefD2D_8Z_Modell` for the basic 3 state space and the standard 8 state space model, respectively.

writes the ODEs and observation equations into the *model.def* file automatically. Furthermore, to improve the numerical condition when estimating parameters, we normalize all free parameters (except initial values and errors) by multiplying them with their nominal value. For example, if the eigenfrequency is to be estimated, we substitute every `Omega0X` in any equation with `(2.0647*Omega0X)`. This way, MATLAB ought to identify every free parameter to 1.

Some of the observables ($\ddot{x}_T$ for the 3 state space and $\ddot{x}_T, \ddot{y}_T$ for the 8 state space model) are an acceleration, i. e. only the derivative of a state. In this case, we need to describe the observable by repeating the complete ODE. Please note that the free parameters need to be normalized in the observation equation as described above, too.

3.3.2 Parameter, Initial State and Error estimation

In this section, the D2D specific settings for parameter estimation are discussed. Thereby, we distinguish between 'actual' parameters (e. g. the air density ϱ or eigenfrequency ω_0), initial values of the dynamic states and errors. Please remember that D2D interprets every parameter not defined numerically in the CONDITIONS section of the *model.def* file as a free parameter that is to be estimated, including initial values and errors defined in the ERRORS section.

When estimating free parameters, one's aim is always to minimize a cost function. D2D uses the objective function (3.10) introduced with the profile likelihood approach,

$$\chi^2(\boldsymbol{\theta}) = \sum_{k=1}^{r}\sum_{l=1}^{d}\left(\frac{y_{kl}^D - y_k(\boldsymbol{\theta}, t_l)}{\sigma_{kl}^D}\right)^2 . \tag{3.10}$$

However, this function is only used as an alternative for the original likelihood function for normally distributed measurement noise,

$$L(\boldsymbol{y}^D|\boldsymbol{\theta}) = \prod_{k=1}^{r}\prod_{l=1}^{d}\frac{1}{\sqrt{2\pi}\sigma_{kl}^D}\exp\left(-\frac{1}{2}\left(\frac{y_{kl}^D - y_k(\boldsymbol{\theta}, t_l)}{\sigma_{kl}^D}\right)^2\right) .$$

By taking the logarithm of the likelihood and multiplying it with -2, we obtain

$$\begin{aligned}
-2\log(L(\boldsymbol{y}^D|\boldsymbol{\theta})) &= \sum_{k=1}^{r}\sum_{l=1}^{d} -2\log\left(\frac{1}{\sqrt{2\pi}\sigma_{kl}^D}\right) - 2\log\left(\exp\left(-\frac{1}{2}\left(\frac{y_{kl}^D - y_k(\boldsymbol{\theta}, t_l)}{\sigma_{kl}^D}\right)^2\right)\right) \\
&= \sum_{k=1}^{r}\sum_{l=1}^{d}\log(2\pi) + 2\log\left(\sigma_{kl}^D\right) + \left(\frac{y_{kl}^D - y_k(\boldsymbol{\theta}, t_l)}{\sigma_{kl}^D}\right)^2 \\
&= \underbrace{r\cdot d\cdot\log(2\pi)}_{\text{const}} + \underbrace{\sum_{k=1}^{r}\sum_{l=1}^{d}2\log\left(\sigma_{kl}^D\right)}_{\text{const if every } \sigma_{kl}^D \text{ is const}} + \underbrace{\sum_{k=1}^{r}\sum_{l=1}^{d}\left(\frac{y_{kl}^D - y_k(\boldsymbol{\theta}, t_l)}{\sigma_{kl}^D}\right)^2}_{\chi^2(\boldsymbol{\theta})} .
\end{aligned} \tag{3.11}$$

Hence, only if the measurement noise for every observable and data point is known and constant, instead of maximizing the likelihood $L(\boldsymbol{y}^D|\boldsymbol{\theta})$ we can minimize $\chi^2(\boldsymbol{\theta})$. And only with the $\chi^2(\boldsymbol{\theta})$-function we can derive confidence intervals from the χ^2-distribution.

Moreover, this short digression is necessary to understand a particularity of the error estimation. If the measurement errors were estimated by minimizing Eq. (3.10), χ^2 would be minimized to 0 by estimating $\sigma^D_{kl} \to \infty$. Thus, the (log-)likelihood function itself is used to estimate the errors along with the other free parameters and initial values, i. e. Eq. (3.11) is minimized. Once the maximum likelihood estimates are found, the error estimations are fixed. Afterwards, the $\chi^2(\boldsymbol{\theta})$-function can be used to derive confidence intervals.

We assume the errors to be *absolute errors*. Alternatively, relative errors could be used (additionally), too, but absolute errors reflect measurement noise more properly. Furthermore, D2D assumes every measurement error to be constant over time, i. e. $\sigma^D_{kj} = \sigma^D_{ki} = \sigma^D_k$. Otherwise, $r \cdot d$ error parameters had to be estimated.

Beside the distinction between errors and other parameters, it is important to understand the possible settings in D2D for their estimation. As default optimization method, D2D uses the MATLAB function `lsqnonlin`, since it has been shown in [40] that multi-start deterministic methods using the sensitivity equations for the calculation of derivatives have significant performance advances in test cases like ours. However, other optimization methods can be selected by setting a value in the global variable `ar`, e. g. `fmincon` with `ar.config.optimizer = 2`.[10]

Furthermore, the optimization can either be done on a log10 or linear scale. Optimizing parameters on a log10 scale can be numerically preferable, especially if the magnitude of the parameter is unknown. However, the parameter must be positive. Therefore, we estimate only the error parameters on a log10 scale. As for the model parameters, we know their nominal value and have normalized them to 1. Hence, we fit them on a linear scale.

Next to the scale, D2D assumes a prior distribution. The distribution consists of a lower bound, upper bound and the type (uniform or Gaussian). If a Gaussian distribution is chosen, its mean and standard deviation can be set, too. The lower and upper bounds are actual bounds, i. e. the parameter cannot be estimated below or above. Additionally, a parameter can be set constant, i. e. if it should not be fitted (or varied during PLE / PPL) furthermore.

All these settings can be configured either directly in the `ar` structure or with the function

`arSetPars(pLabel_or_ar, p, qFit, qLog10, lb, ub, type, meanp, stdp)`.

Table 3.4 summarizes the possible settings and our choices, depending on the type of parameter.

The D2D framework also provides a very handy function `arSetParsPattern(pattern, ...)`, which applies `arSetPars(...)` on all free parameters containing the string `pattern` in their names, as is exemplary shown in listing 3.2.

Listing 3.2: Setting the fitting options for v_w and all error parameters

```
arSetParsPattern('Wind1VelX', 1, 1, 0, -2, 15);
arSetParsPattern('_abs', 0, 1, 1, -5, 3);
```

[10] See `https://github.com/Data2Dynamics/d2d/wiki/Optimization%20algorithms%20available%20in%20the%20d2d-framework` or type `ar.config.optimizers` for a full overview.

Table 3.4.: Overview of chosen settings for model parameters, initial values, absolute errors and wind velocity, if identified as a constant parameter. $x_i(t_0)$ refers to the true initial value of the corresponding state.

Setting	Description	Model	Initial	Errors	v_w
`p`	prior value	0.9	0	10^0	1
`qFit`	0 = constant	1	1	1 before fitting	1
	1 = to be estimated			0 after fitting	
`qLog10`	0 = linear	0	0	1	0
	1 = log10 scale				
`lb`	lower bound (hard)	0	$-10 \cdot \lvert x_i(t_0) + 10 \rvert$	10^{-5}	-2
`ub`	upper bound (hard)	2	$+10 \cdot \lvert x_i(t_0) + 10 \rvert$	10^3	15
`type`	0 = Uniform	0	0	0	0
	1 = Gaussian				
`meanp`	Gaussian mean	-	-	-	-
`stdp`	Gaussian SD	-	-	-	-

It should be noticed that the objective function in (3.10) and all conclusions drawn from it in the profile likelihood theory assume the errors to come only from measurement noise. However, in our case most of the errors are due to modeling errors, at least if we compare a simplified model to more complex data (see Section 3.4 for more details). Therefore, we let them be estimated like the other parameters. Furthermore, by fitting the errors, we normalize the observables in the objective function (3.10). Otherwise, e. g. n_g with a mean of $\approx 1000\,\mathrm{rpm}$ would dominate the objective function compared to $\ddot{x}_T$ with a maximum of $\approx 1\,\mathrm{m/s^2}$.

The actual fitting can be done with the function `arFit`. However, to increase the probability to find the global minimum, we use `arFitLHS(n, ...)`, which basically starts `arFit n` times with different starting values and sets the parameter values to the best fit found.

3.3.3 (Prediction) Profile Likelihood

Please note that the theoretical background for the profile likelihood (PL or PLE) and the prediction profile likelihood (PPL) is discussed in Section 2.2.1 on page 14 and in Section 2.2.2 on page 27, respectively. This sections focus lies on the application of both methods in the D2D framework.

Before applying the PLE (or PPL) method, it is recommended to fit the parameters as described in the section above to make sure the global minimum is found. Then, only 2 function calls are mandatory. Remember that the aim of PLE is to find confidence intervals for every parameter, i. e. an interval in which the true value of the parameter lies with a probability of $1 - \alpha =$

$1 - 0.05 = 95\,\%$.[11] However, first tests showed that in our use case, already small changes in the parameters cause substantial increments in the objective functions value. Therefore, it might happen that the confidence interval consists only of one point, i. e. the optimum found before. To prevent that, we adjust the step sizes (minimum and especially maximum) and decrease the integration tolerances (both relative and absolute).[12]

Furthermore, we have to hold the estimated error parameters constant. Otherwise, they would be adjusted for every change in a parameter and the use of the objective function would be illegitimate as described in Section 3.3.2. Obviously, the PLE itself must not be applied to the errors either. Additionally, we want the PLE to break if it hits the lower or upper parameter bound. Listing 3.3 shows how to apply the described adjustments and start the PLE method.

Listing 3.3: Initializing, preparing and processing PLE in MATLAB.

```
arPLEInit;                  % Initialize PLE
ar.config.rtol = 1e-10;     % Set relative tolerance
ar.config.atol = 1e-10;     % Set absolute tolerance

% Set all error parameters constant (3rd argument = 0)
arSetParsPattern('_abs', [], 0, 1, -5, 3);

% Determine all non-error parameters
p_free = ar.pLabel(cellfun(@isempty, strfind(ar.pLabel, '_abs')));

% Apply PLE
% ple(parameters, samplesize, ..., maxstepsize, minstepsize, →
   ←breakon_lb, breakon_ub)
ple(p_free, 200, [], 1e-2, 1e-8, 1, 1);
```

As a result, we get a qualitative assessment about a parameters identifiability by the corresponding PLE curve, i. e. whether it is structurally and/or practically identifiable or non-identifiable. Additionally, the confidence intervals are quantitative assessments. However, when examining the resulting confidence intervals, we need to keep in mind that we usually have modeling errors we cannot neglect. Therefore, we expect the width of a confidence interval to be a measure of the parameters identifiability, which we will verify by simulation in Section 4.5.1. But its 95 % likelihood may not be true.

The second method we are applying is PPL. Unfortunately, its implementation in D2D is much more fragile than PLE's. Its aim is to assess the observability of a dynamic quantity, basically by determining confidence intervals over the time course. The width of the confidence intervals

[11] $\alpha = 0.05$ is usually chosen and the default value in D2D, but could be altered to any other value $\in (0, 1)$.

[12] Please refer the online wiki for more details on tolerances, `https://github.com/Data2Dynamics/d2d/wiki/Integration%20tolerances%20and%20modification%20possibilities`

then are a quantitative measure for its observability. (Again, just as for PLE, due to modeling errors their probability conclusion may not be valid.) The dynamic quantity may either be a state or a (non-measured) observable.

As for D2D, we need to set a few options, partly to ensure its processing without error. The function call is given by

```
doPPL(m, c, ix, t, takeY, options)
```

and Table 3.5 summarizes the possible and chosen arguments. Normally, the function `doPPL`

Table 3.5.: Overview of (chosen) arguments for `doPPL`. The last argument `options` is optional and may contain up to 14 settings, of which three are changed by us. The value in curly brackets {} is the default setting. For more details, see the documentation of `PPL_options`.

Argument	Description	Our setting
`m`	nr. of model to use	1
`c`	nr. of condition to use	1
`ix`	nrs. of states/observables to use	[], i. e. all
`t`	time points for which PPL is to compute	every 10th
`takeY`	0 = for states, 1 = for observables	0
`options:`		
`'onlyProfile'`	no interpolation between points {false}	true
`'doPPL'`	comp. prediction profile, not validation profile {false}	true
`'rel_increas'`	Changes in x during computation {0.15}	0.3

would interpolate PPL confidence intervals between the computed points given by `t`. Since these interpolated values are not valid even theoretically, we choose not to use them. Since we are interested in the prediction confidence intervals and not in the validation confidence intervals, the optional argument `'doPPL'` is set to `'true'`. Furthermore, the setting `'rel_increas'` has to be increased. Otherwise, with no overlaid measurement noise, we might get too narrow bounds which would lead to simulation errors. Listing 3.4 shows the corresponding MATLAB code.

Listing 3.4: Calling the PPL function in MATLAB.

```
options = struct('onlyProfile', true, 'doPPL', true, 'rel_increase',...
    ... 0.3);
t_vec = [wtsim.t(n_start):10*settings.data_dt{1}:wtsim.t(n_start+...
    ...n_max*t_dif-1)];
doPPL(1,1,[],t_vec,0, options);
ar.config.ploterrors=-1; %Plot prediction bands
arPlot2;
```

3.4 Scenario Variations

A big advantage of our implementation in the D2D framework is its flexibility to analyze our models with different settings and circumstances. To do so, a regular MATLAB script calls a function `doD2D(des_settings)` with a structure containing the desired settings as input argument. By combining different settings in a superior structure, we can simulate multiple scenario variations by iterating through this superior structure. This chapter discusses the possible variations. Therefore, Table 3.6 summarizes every field of the input structure and its possible values. Furthermore, every field is discussed shortly in a separate paragraph.

Table 3.6.: Overview of possible field/variable settings to call `doD2D(des_settings)` with

Variable	Possible values
`model`	`'Basic_3Z'`, `'Standard_8Z'`
`data_source`	`'Basic_3Z'`, `'Standard_8Z'`, `'Advanced_21Z'`
`data_scenario`	`'steady'`, `'low_turb'`, `'medium_turb'`, `'high_turb'`
`data_nDataPoints`	every positive integer $< (80{,}000 - \texttt{data\_nStart}) \cdot 0.01/\texttt{data\_dt}$
`data_nStart`	every positive integer $< 80{,}000 - \texttt{data\_nDataPoints} \cdot \texttt{data\_dt}/0.01$
`data_dt`	every manifold of `0.01`
`free_parameters`	every combination of parameters, e. g. `'Omega0X+AirDensity'`
`free_initials`	every combination of states, e. g. `'QD_GeAz+TTDspFA'`
`noise_snr_db`	every positive value, `0` for none
`noise_seed`	every nonnegative integer below 2^{32}
`interpolation`	`@writeSuperSplines`, `@writeSplines` or `@writeSteps`
`inputs`	`'GenTq'`, `'Wind1VelX'` or `'GenTq+Wind1VelX'`
`observables`	every comb. of observables, e. g. `'Azimuth+NcIMUTAys'`, or `'all'`
`doPLE`	`0` or 1
`doPPL`	`0` or 1
`nFitLHS`	every positive integer value

`Model`

Both wind turbine models introduced in Section 3.1 can be used and analyzed. The possible settings are `'Basic_3Z'` and `'Standard_8Z'` for the basic model with 3 states and the standard model with 8 states, respectively.

`data_source`

When analyzing our models identifiability or observability, we have to distinguish between their theoretical/mathematical assessment and their practical relevance 'in field'. Therefore, we use

different data sources. For the theoretical assessment, we 'feed' the D2D framework with data from previous simulations of the model itself, i. e. the basic or standard model. To assess how well identifiable or observable the models would be in reality, we use data from an advanced model with 21 states and assume them to reflect real on-site data.

`data_scenario`

As has been discussed in chapter 2, the identifiability and observability of a nonlinear system depend on the system's stimulation (input). Therefore, wind scenarios with different turbulences are respected. The turbulence is defined as the wind velocity's standard deviation divided by its mean, $\iota = \sigma(v_\mathrm{w})^2/\overline{v}_\mathrm{w}$. Next to a constant wind velocity (`'steady'`), we can choose between three different turbulence intensities, i. e. `'low_turb'` ($\iota = 0.0458$), `'medium_turb'` ($\iota = 0.0925$) and `'high_turb'` ($\iota = 0.1887$). Thereby, the mean of the wind velocity is always $\overline{v}_\mathrm{w} = 9\,\frac{\mathrm{m}}{\mathrm{s}}$.

`data_nDataPoints`

This field states the number of data points to be used. The data sets we use have a length of 80,000. Furthermore, we need to consider the first point of the data set we take and whether or not we use every point. Since in general we simulate much less than the maximum of 80,000 points, the maximum allowed value is not exceeded anyway.

`data_nStart`

This field states the first point of the data set to use. Furthermore, we need to consider the overall amount of data points to be taken and whether or not we use every point. Again, usually the maximum allowed value is not exceeded.

`data_dt`

In practice, the sampling frequencies of sensors are limited. That means we need to take the time difference between two data points into account. The data sets we use have a time increment of $\Delta\mathrm{T} = 10\,\mathrm{ms}$, i. e. 0.01 s. Therefore, we can set the desired $\Delta\mathrm{T}$ to any (whole-numbered) manifold of 0.01.

`free_parameters`

The key functionality of D2D is the handling of unknown parameters. In our case, we know all parameters the systems have been simulated with originally. Our implementation does define every parameter's value in the corresponding section of the *model.def* file, if not told otherwise. To interpret a parameter as unknown, it has to be stated in this field. Multiple parameters can be defined as unknown by combining them with a + to a single string. The wind velocity is

estimated as a constant parameter automatically if not defined as a known input (see paragraph `inputs` below).

`free_initials`

Similar to `free_parameters` , the states whose initial values should not be known can be defined here. The corresponding states are defined by a single string containing the states names, again combined by a +. Please note that the addition of `init_` is *not* necessary.

`noise_snr_db`

In reality, measurements are always overlaid by noise. Therefore, we can implement artificial noise with a desired signal to noise ratio, given in dB. Any dB value can be set. A `0` indicates that no noise shall be used.

`noise_seed`

For reproducibility and to test the impact of different noise implementations, the seed of which MATLAB produces the noise from can be set. For more information the reader is referred to the online documentation.[13]

`interpolation`

As has been discussed in Section 3.3.1, three different interpolation methods have been used during this work to represent the input data. This field expects a function handle to one of these methods. However, since it is only advantageous, `@writeSuperSplines` will be used solely.

`inputs`

Here, the known inputs are defined, again as a single string containing the input names combined by a `'+'`. If the standard model with 8 states is chosen, the three blade pitches are defined as known inputs automatically.

`observables`

Again, all measured observables are given as a single string containing the corresponding names combined by a `'+'`. Alternatively, `'all'` states that all available observables are measured.

`doPLE`

Simple flag to define whether the PLE is computed or not (`1` = yes, `0` = no).

[13] For example `https://de.mathworks.com/help/matlab/math/generate-random-numbers-that-are-repeatable.html`

`doPPL`

Simple flag to define whether the PPL is computed or not (`1` = yes, `0` = no).

`nFitLHS`

Defines the number of starting points for the first fitting of the free parameters, i. e. first input argument `n` of `arFitLHS(n, ...)`.

4 Simulation Results

In this chapter, the results obtained by simulating our models in the D2D framework are presented. First, an influence assessment of the chosen data representation is given. Secondly, a qualitative overview summarizes the identifiability and observability of various settings. Then, a deeper consideration of the identifiability of different parameter settings is given, followed by similar observability analyses.

4.1 Influence of Data Quality and Quantity

In Section 3.4, the possible scenario variations have been discussed. However, infinite combinations are possible, and even restricting to only reasonable options and differences in quantitative factors would still lead to multiple thousands of scenarios. Hence, we need to constrain the number of factors that are to be analyzed more deeply. Therefore, in this section, we evaluate the influence of the chosen data in terms of its amount, time gaps, noise and its content, i. e. the wind turbulence.

4.1.1 Amount of Data Points

For every evaluation in this section, we again need to restrain the possible settings. For a reduction in run time, we choose to use only the basic model with three states but feed it with both data from a simulation of itself and from the advanced model with 21 states, which is assumed to reflect real data. Furthermore, to test how much data is necessary at most, we choose the high turbulence wind scenario. As free parameters, the eigenfrequency ω_0 and the air density ϱ are chosen, the initial values of the states are assumed to be known. Table 4.1 summarizes the general simulation settings for all tests in this section. As quality measure q we use the averaged sum of the relative errors, i. e.

$$q = \frac{1}{n} \sum_{i}^{n_p} \left(\frac{p_{i,\text{est}}}{p_{i,\text{true}}} - 1 \right) .$$

In the bar plots given below, we also show the parameter estimate's share of q. If only two parameters are fitted like in this section, the actual relative error of a parameter fit is twice as big as represented in the plot.

Figure 4.1 shows q for various amounts of data points for no noise and a signal to noise ratio (SNR) of 30 dB, which is equivalent to $\approx 5\,\%$ measurement noise we can assume for real on site data.

Table 4.1.: General simulation settings for the tests presented in this section.

Variable	Possible values
model	'Basic_3Z'
data_source	'Basic_3Z', 'Advanced_21Z'
data_scenario	'high_turb'
data_nDataPoints	200 or varied
data_nStart	1000
data_dt	0.1 or varied
free_parameters	'Omega0X+AirDensity'
free_initials	'', i. e. none
noise_snr_db	0, 30 or varied
noise_seed	1
interpolation	@writeSuperSplines
inputs	'GenTq+Wind1VelX'
observables	'all'
doPLE	1
doPPL	0 or 1
nFitLHS	50

As one can see, for $n_{\text{Data}} \geq 200$ data points the errors are negligibly small, even for data from the advanced model. For 400 or more data points, it is noticeable that the errors are nearly constant, but vary little to both sides. However, the deviations are so small that they can be assumed to be from numerical issues or just unlucky chosen data, but no significant trend to a decline or increase of estimation accuracy coming with an increase of data points is deducible.

Therefore, to keep computational costs in further simulations as low as possible, we restrain the number of data points to 200.

4.1.2 Intervals of Data Points

Next to the amount of data points, we need to take a look at the influence of the sampling frequency. The minimum time interval we can choose is $\Delta\text{T} = 0.01\,\text{s}$. However, it might be advantageous to choose a lower sampling frequency.

Figure 4.2 shows the averaged summed error for the basic model with data from itself and from the advanced model with 21 states, both times for no noise and a SNR of 30 dB. As one can see, the results are not as clear as for the amount of data points before. For the data from the basic model itself, no clear trend is deducible. With no noise the minimum is at $\Delta T = 0.1\,\text{s}$, but for the noisy measurements it is the maximum error at the same time. However, for the 'real' data, the minima for both with and without noise lie at $\Delta T = 0.1\,\text{s}$, too.

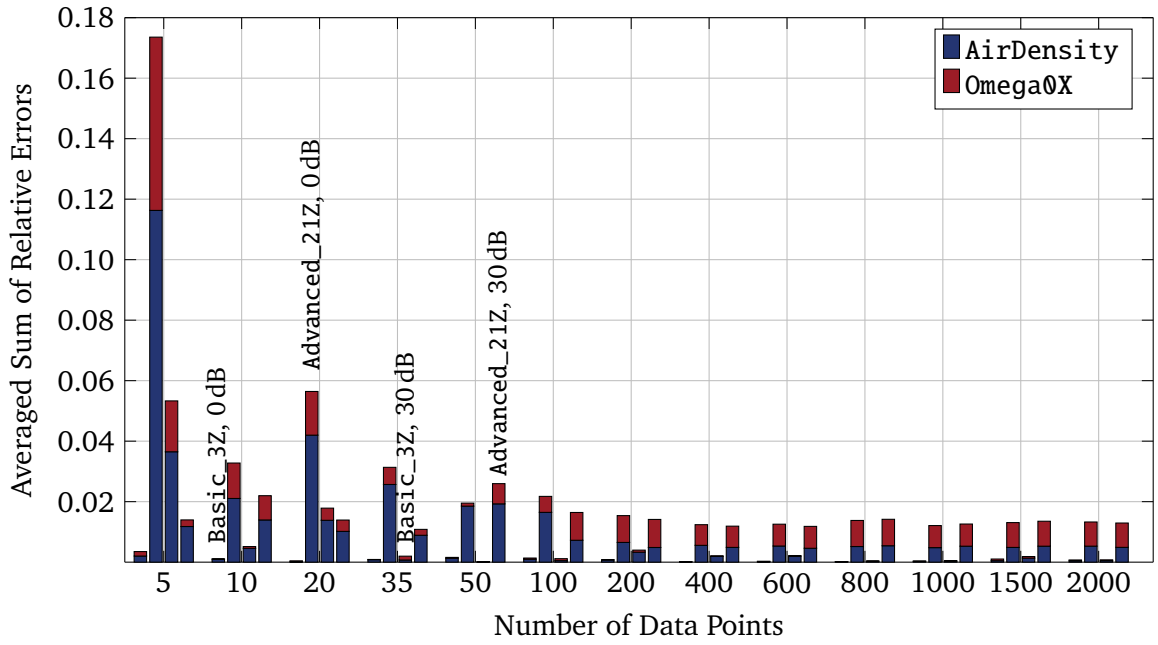

Figure 4.1.: Comparison of q for different numbers of data points using the basic model. The four different scenarios are the combinations of basic and advanced model data with no noise and a SNR of 30 dB.

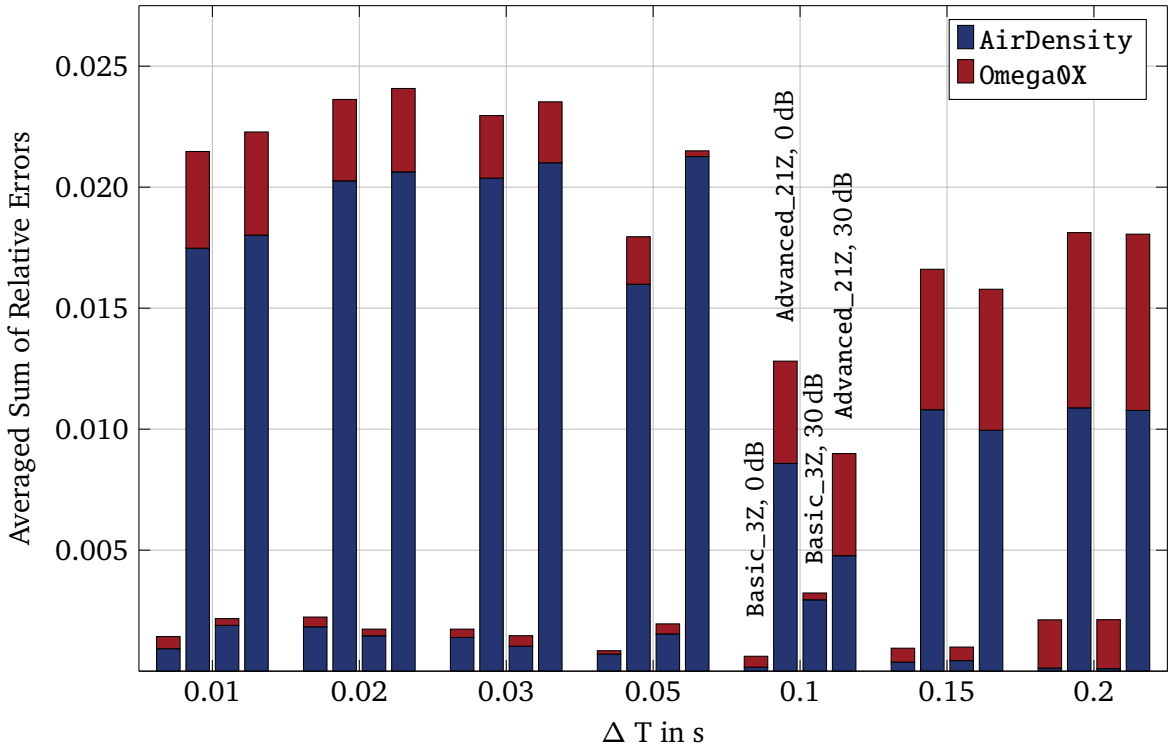

Figure 4.2.: Comparison of q for different time intervals using the basic model. The four different scenarios are the combination of the basic and advanced model data with no noise and a SNR of 30 dB.

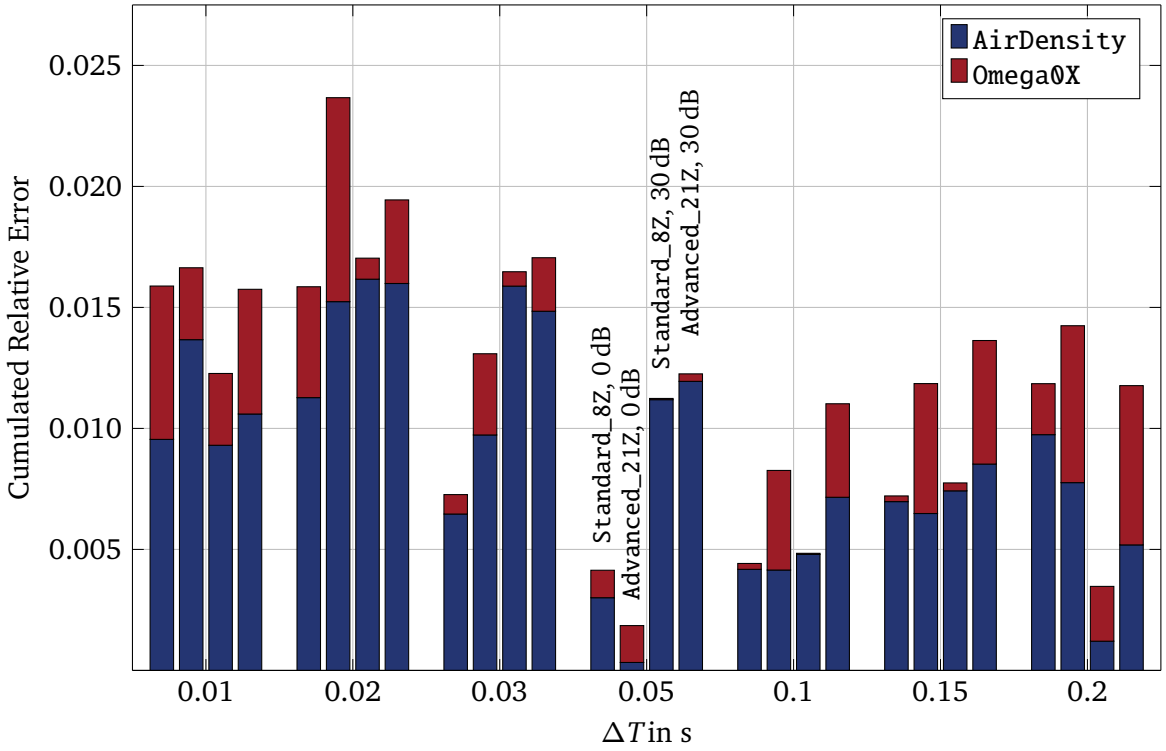

Figure 4.3.: Comparison of q for different time intervals using the standard model. The four different scenarios are the combination of the standard and advanced model data with no noise and a SNR of 30 dB.

Therefore, we apply the same tests at the standard model, feeding it with its own and the real data. The results are shown in Figure 4.4 and look very similar. Again, the minimum at least for the noisy data from the advanced model lies at $\Delta T = 0.1\,\mathrm{s}$. Without noise, the minimum for the data from the advanced model lies at $\Delta T = 0.05\,\mathrm{s}$. For the data from the standard model itself, the minima lie at $\Delta T = 0.05\,\mathrm{s}$ and $\Delta T = 0.2\,\mathrm{s}$ for without and with noise, respectively.

Summarizing, at $\Delta T = 0.1\,\mathrm{s}$, the errors are small for all combinations and minimal for the most interesting setting with the real and noisy data. Furthermore, a sampling frequency of $f_\mathrm{s} = \frac{1}{0.1\,\mathrm{s}} = 10\,\mathrm{Hz}$ is realistic. Therefore, that is the time interval setting we choose for further investigations.

4.1.3 Noise comparisons

To obtain a first assessment of the effect of noise, we again simulate the basic model with the general settings given in Table 4.1 and for various SNR levels. The results are given in Figure 4.4. As one can see, using data from the basic model itself, the errors increase with the noise. Only at a SNR of 10 dB the errors have decreased, which is an exceptional behavior and may be due to an 'unlucky' choice of noise. At the realistic noise level of 30 dB, the average error per parameter is still $< 1\,\%$.

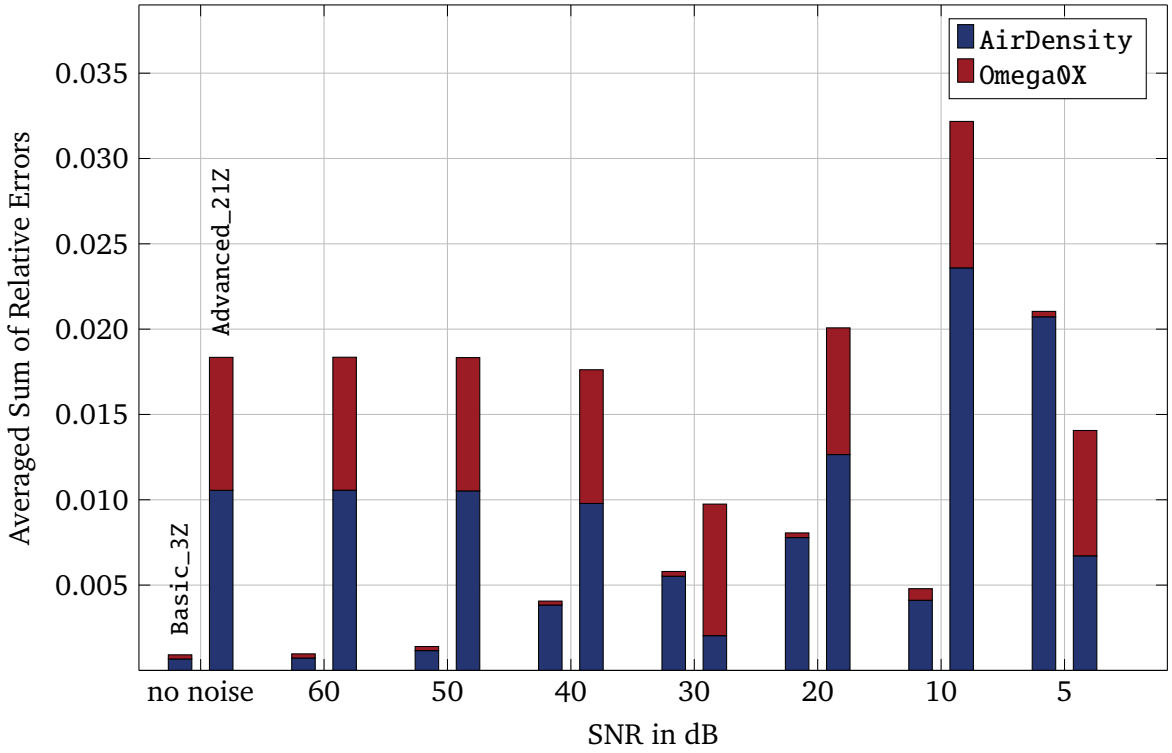

Figure 4.4.: Comparison of q for different SNR levels using the basic model with $\Delta T = 0.1\,\mathrm{s}$.

As for data from the advanced model, low noise levels do not seem to have a significant influence. For a SNR of 30 dB, the error is even lower than for no noise, but increases again for 20 and 10 dB. However, it declines again for 5 dB. All in all, solely for a SNR of 10 dB the average error per parameter is higher than 2.01 %. This shows that the profile likelihood method is designed to handle Gaussian distributed noise and has greater difficulties with the significant model error between the basic and advanced model. If real data is used, the noise seems to have nearly no significant effect at all.

4.2 Qualitative Overview ...

Since we have evaluated the influence of the data quality and quantity, we now simulate our models for various combinations of unknown parameters, measured outputs and data sources. In this section, we first give a qualitative overview of the identifiability of these scenarios, followed by an equivalent overview of the models' observability. For completeness, we summarize the general simulation settings in Table 4.2.

Please note that, in contrast to the data tests in the section before, we assume the initial values to be unknown. We want to evaluate the models qualitative identifiability and observability under realistic circumstances. Therefore, we cannot expect all state variable initials to be known, only those which comply with a measured output. If so, the output measurements are committed anyway, i. e. the initial value is identifiable or known directly. Furthermore, we decreased the

Table 4.2.: General simulation settings for the qualitative identifiability and observability assessment.

Variable	Possible values
model	varied
data_source	varied
data_scenario	'high_turb'
data_nDataPoints	200
data_nStart	1000
data_dt	0.1
free_parameters	varied
free_initials	'all'
noise_snr_db	30
noise_seed	1
interpolation	@writeSuperSplines
inputs	varied
observables	varied
doPLE	1
doPPL	0
nFitLHS	25

number of starting values for the fitting function from 50 to 25 to reduce computational costs, which is legitimate since we do also calculate the PLE.

4.2.1 ... of identifiable / non-identifiable settings

In this section, we analyze the structural and practical identifiability for various scenarios. First, the basic model with three states is considered, followed by the standard model with eight states.

The formal definitions of both structural and practical identifiability are given in the paragraph Profile Likelihood Approach on page 14. However, during the evaluation of our results, it turned out that their direct application is not purposeful. That is, there are many cases in which the parameter would be practically identifiable according to the definition, i. e. you do get a limited confidence interval (CI). However, the width of the confidence is often many times greater than the parameter value itself, e. g. an interval $[-30, 30]$ for a parameter normalized to 1.

Therefore, we differ from the original definition of practical identifiability. We assume a parameter to be non-identifiable if its confidence interval is either unlimited or bigger than 50 % of the true parameter value $\theta_{i,\text{true}}$. If it is smaller than 50 % but the best parameter fit differs more

than 10 % of the true value, we assume it to be *partly* practically identifiable, which we denote with a '∗p'. Table 4.3 summarizes the connotation used in the following sections.

Table 4.3.: Connotation of structural and practical identifiability.

Sign	Meaning	Explanation
+s	structurally identifiable	χ^2_{PL} changes over θ_i
−s	structurally non-identifiable	χ^2_{PL} does not change over θ_i
+p	practically identifiable	limited CI $\leq 0.5 \cdot \theta_{i,\mathrm{true}}$ and $\lvert\frac{\theta_{i,\mathrm{est}}}{\theta_{i,\mathrm{true}}}\rvert - 1 \leq 0.1$
∗p	partly practically identifiable	limited CI $\leq 0.5 \cdot \theta_{i,\mathrm{true}}$ but $\lvert\frac{\theta_{i,\mathrm{est}}}{\theta_{i,\mathrm{true}}}\rvert - 1 > 0.1$
−p	practically non-identifiable	CI unlimited or $> 0.5 \cdot \theta_{i,\mathrm{true}}$

Basic Model

The results for the basic model are displayed in Tables 4.4, 4.5, 4.6 and 4.7, which comply to the combinations of both data sources (the basic model itself and the advanced model) and whether the wind is assumed to be a known input (with high turbulances) or an unkown parameter (with a constant value). The corresponding settings are given in every table's caption.

Furthermore, we simulated three different constellations of free parameters (ϱ, ω_0 or ϱ and ω_0) and three different constellations of measurements (n_g and $\ddot{x}_T$, n_g or $\ddot{x}_T$). The structural and practical identifiability of a parameter are indicated by a sign (+ or −) and an s or p, respectively. For example, when both generator speed and the tower acceleration are measured, both eigenfrequency and air density are structural and practically identifiable. Therefore, the corresponding field in Table 4.4 states +s+p +s+p. If three parameters are identified, the statement is abbreviated by using only signs, e. g. ++ +− .

Table 4.4.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Basic_3Z`, wind scenario: `high_turb`

Measured: / Identified:	`GenSpeed+NcIMUTAxs`	`GenSpeed`	`NcIMUTAxs`
`Omega0X`	+s+p	+s−p	+s+p
`Airdensity`	+s+p	+s+p	+s+p
`AirDensity+Omega0X`	+s+p +s+p	+s+p +s−p	+s+p +s+p

As can be seen, both parameters as well as the wind velocity are structurally identifiable for every constellation analyzed here. However, the practical identifiability varies. Let us first assume the wind velocity to be known. Then, if the generator speed as well as the tower

Table 4.5.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.

Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X	+s+p	+s∗p	+s+p
Airdensity	+s+p	+s+p	+s∗p
AirDensity+Omega0X	+s+p +s+p	+s+p +s∗p	+s∗p +s+p

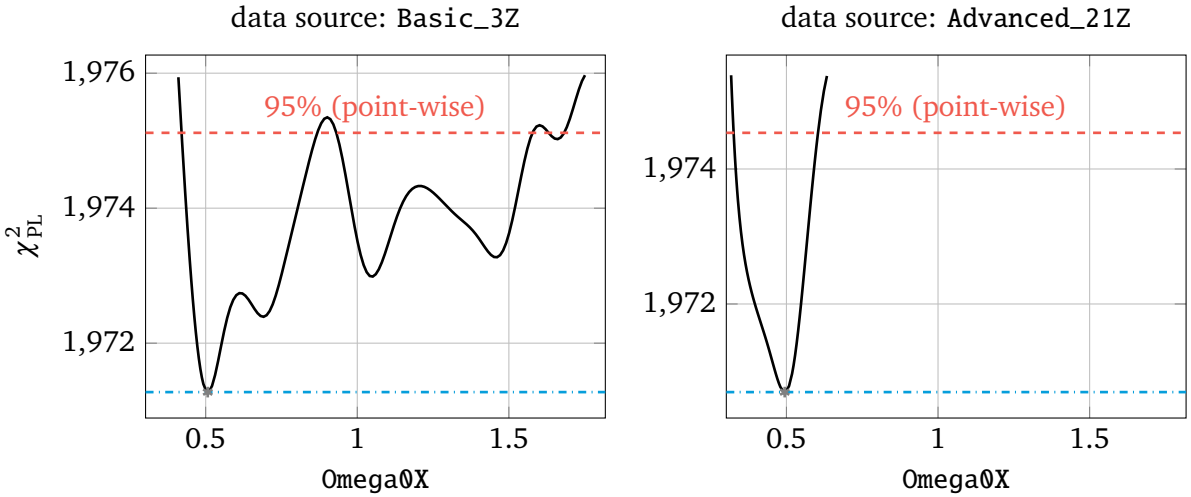

Figure 4.5.: Profile likelihood progressions for the eigenfrequency. For data from the basic model itself, the profile likelihood hits the χ^2-limit multiple times and is therefore practically non-identifiable. For data from the advanced model, the confidence interval width is small enough for the eigenfrequency to be at least partly practically identifiable.

Model: `Basic_3Z`, data source: varied, wind scenario: `high_turb`, measured outputs: `GenSpeed`

acceleration is measured, both air density and eigenfrequency are practically identifiable. If only the generator speed is measured, the eigenfrequency loses its practical identifiability. However, it is noteworthy that, at least with our criteria, it is still partly practically identifiable when data from the advanced model is used, but practically non-identifiable when the original data from the model itself is used. Figure 4.5 shows the profile likelihood progressions for both constellations. It seems that the artificial measurement noise we assumed is outweighed by the model error from the advanced model data, hence leading to a less fluctuating profile likelihood curve in this case. When the tower acceleration is measured instead of the generator speed, the eigenfrequency is practically identifiable in any case. In return, the air density is only partly practically identifiable when data from the advance model is used.

If the wind velocity is identified as a constant parameter, too, the overall practical identifiability suffers drastically. More precisely, air density and wind velocity together are never practically

Table 4.6.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Basic_3Z`, wind scenario: `steady`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X+Wind1VelX	++ ++	+− +−	++ +∗
Airdensity+Wind1VelX	+∗ ++	+− +−	+∗ +∗
AirDensity+Omega0X+Wind1VelX	+∗ ++ ++	+− +− +−	+∗ ++ +∗

Table 4.7.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `steady`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X+Wind1VelX	++ ++	+− +−	++ +−
Airdensity+Wind1VelX	+∗ +∗	+− +−	+∗ +−
AirDensity+Omega0X+Wind1VelX	+∗ ++ +∗	+− +− +−	+− ++ +−

identifiable. If both outputs are measured, at least the wind velocity is practically identifiable when data from the basic model itself is used, but only partly practically identifiable with data from the advanced model.
The air density is partly practical identifiable if the tower acceleration is measured, except if data from the advanced model is used and the eigenfrequency is identified at the same time - then, the generator speed needs to be measured, too. If the generator speed is the only output, none of the parameters are practically identifiable for any constellation, not even partly.

Concluding, the practical identification of the wind velocity together with the air density is barely possible. That seems reasonable considering that they appear nearly only as a product in Eqs. (3.1) and (3.2), except for the blade tip speed ratio λ in Eq. (3.3).

Standard Model

Using the standard model with its maximum of four outputs, 15 different measurement constellations are possible, of which five are chosen here. Otherwise, the same constellations as for the basic model have been analyzed, so that the models can be compared to each other. Tables 4.8 and 4.9 give the results with the wind as a known input, Tables 4.10 and 4.11 those when the wind velocity is identified as a constant parameter.

Table 4.8.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.

Model: Standard_8Z, data source: Standard_8Z, wind scenario: high_turb

Measured: / Identified:	GenSpeed+Azimuth +NcIMUTAxs+NcIMUTAys	GenSpeed	NcIMUTAxs
Omega0X	+s+p	+s∗p	+s+p
Airdensity	+s+p	+s+p	+s+p
AirDensity+Omega0X	+s+p +s+p	+s+p +s∗p	+s+p +s+p

	GenSpeed+Azimuth	GenSpeed+NcIMUTAxs
Omega0X	+s∗p	+s+p
Airdensity	+s+p	+s+p
AirDensity+Omega0X	+s+p +s∗p	+s+p +s+p

Table 4.9.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.

Model: Standard_8Z, data source: Advanced_21Z, wind scenario: high_turb

Measured: / Identified:	GenSpeed+Azimuth +NcIMUTAxs+NcIMUTAys	GenSpeed	NcIMUTAxs
Omega0X	+s+p	+s∗p	+s+p
Airdensity	+s+p	+s+p	+s∗p
AirDensity+Omega0X	+s+p +s+p	+s+p +s∗p	+s∗p +s+p

	GenSpeed+Azimuth	GenSpeed+NcIMUTAxs
Omega0X	+s∗p	+s+p
Airdensity	+s+p	+s+p
AirDensity+Omega0X	+s+p +s∗p	+s+p +s+p

First, we consider the wind to be the input. Again, as for the basic model, all parameters are structurally identifiable for every constellation. Furthermore, comparing the standard to the basic model for the same measurements, one can see that their identifiability is nearly the same. The only difference is that the eigenfrequency is partly practically identifiable instead of practically non-identifiable for both data sources even when only the generator speed is measured. For the basic model, this was only valid when data from the advanced model was used.

Obviously, all parameters are also practically identifiable when the azimuth angle and the tower acceleration in y-direction is measured additionally to the generator speed and the acceleration in x-direction. However, measuring the azimuth angle does not seem to contribute any information to the practical identifiability, since the eigenfrequency is only partly practically identifiable no matter if the generator speed is measured with or without the azimuth angle.
As can be seen from a comparison of Tables 4.8 and 4.9, the only difference between both data sources is that the air density is only partly practically observable if data from the advanced model is used and only the tower acceleration in x-direction is measured.

Tables 4.10 and 4.11 give the results with the wind being identified as a constant parameter. Comparing the standard with the basic model with data from themselves, i. e. Tables 4.6 and 4.10, there are some notable differences. If only the generator speed is measured, the eigenfrequency is partly practically identifiable for the standard but not for the basic model. However, it becomes non-identifiable again if the azimuth angle is measured at the same time. The air density is even fully practically identifiable if only the generator speed is measured - but only if the eigenfrequency is identified at the same time, otherwise not. It seems that in this case additional freedoms improve the practical identifiability. Again, the air density becomes practically non-identifiable if the azimuth angle is measured additionally.
If only the fore-aft tower acceleration is measured, both air density and wind velocity are partly practically observable for the basic but not for the standard model.

Table 4.10.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.
Model: `Standard_8Z`, data source: `Standard_8Z`, wind scenario: `steady`

Measured: / Identified:	GenSpeed+Azimuth +NcIMUTAxs+NcIMUTAys	GenSpeed	NcIMUTAxs
Omega0X+Wind1VelX	++ ++	+∗ +−	++ +∗
Airdensity+Wind1VelX	+∗ ++	+− +−	+− +−
AirDensity+Omega0X+Wind1VelX	+∗ ++ ++	++ +∗ +−	+− ++ +−

	GenSpeed+Azimuth	GenSpeed+NcIMUTAxs
Omega0X+Wind1VelX	+− +−	++ ++
Airdensity+Wind1VelX	+− +−	+∗ ++
AirDensity+Omega0X+Wind1VelX	+− +− +−	+∗ ++ ++

If we compare both data sources for the standard model, i. e. Tables 4.10 and 4.11, we see that there is nearly no difference regarding the identifiability of the parameters. The only changes are that the wind velocity becomes only partly practically identifiable instead of practically identifiable when identifying all three parameters and measuring all outputs or at least the generator speed together with the fore-aft tower acceleration. And if we solely measure the

Table 4.11.: Overview of structural and practical identifiability, in dependence of the measured outputs and the parameters to be identified.
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `steady`

Measured: / Identified:	GenSpeed+Azimuth +NcIMUTAxs+NcIMUTAys	GenSpeed	NcIMUTAxs
Omega0X+Wind1VelX	++ ++	+∗ +−	++ +−
Airdensity+Wind1VelX	+∗ +∗	+− +−	+− +−
AirDensity+Omega0X+Wind1VelX	+∗ +++∗	++ +∗ +−	+− ++ +−

	GenSpeed+Azimuth	GenSpeed+NcIMUTAxs
Omega0X+Wind1VelX	+− +−	++ ++
Airdensity+Wind1VelX	+− +−	+∗ +∗
AirDensity+Omega0X+Wind1VelX	+− +− +−	+∗ ++ +∗

tower acceleration and identify the wind velocity together with the eigenfrequency, it becomes practically non-identifiable instead of partly identifiable. Hence, we refrain from comparing the basic and the standard model for data from the advanced model.

4.2.2 ... of observable / unobservable settings

After assessing the qualitative identifiability of our models for different conditions, we do the same for the observability of their states. By recalling the definition of observability, it becomes clear that a system's observability depends on whether the initial state vector $\boldsymbol{x}_0$ can be determined or not. However, the initial value of a single state $x_{i,0}$ is basically a parameter. Therefore, we can determine the qualitative observability of a state by the identifiability of its initial value.

Furthermore, to formalize this approach, we introduce two definitions. First, the structural observability of a state is defined analogous to the structural identifiability of a parameter.

Definition 4.1 *A state x_i is* structurally unobservable*, if the likelihood-based confidence interval of its initial value $x_{i,0}$ is infinite, $[-\infty, +\infty]$, and its profile likelihood has no unique minimum, but is flat over all x_i. Otherwise, it is* structurally observable.

The term *practical observability* has already been suggested by *Kreuz et al.* in [25], but has not been formalized. Therefore, we secondly define it as follows.

Definition 4.2 *A state x_i is* practically unobservable*, if the likelihood-based confidence region of its initial value is infinitely extended in increasing and/or decreasing direction of x_i, although the likelihood has a unique minimum for this parameter.*

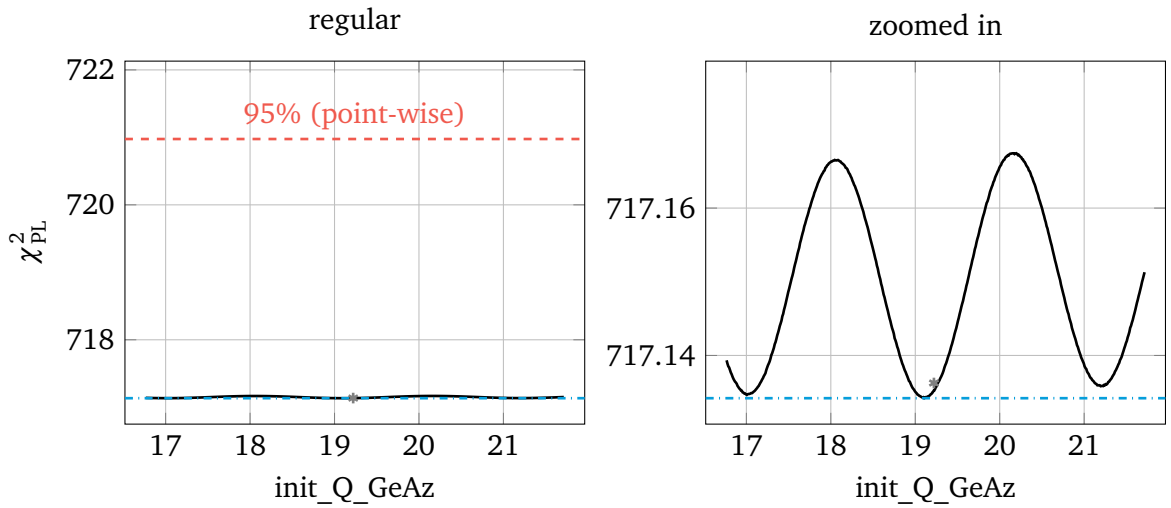

Figure 4.6.: Exemplary profile likelihood progression for the azimuth angle if not measured. Left: Regular plot, right: vertically zoomed in. The best fit indicated by the asterisk is not precisely at the minium of χ^2_{PL} due to numerical issues.
Model: `Standard_8Z`, data source: `Standard_8Z`, wind scenario: `high_turb`, measured outputs: `GenSpeed+NcIMUTAxs`

Unfortunately, in practice the same problems occurred as for the identifiability of parameters. Sometimes, a limited CI can be found but is so large that it mainly contains unreasonable values. Therefore, when assessing a state's observability, the same rules as in Table 4.3 are applied, only with an initial value $x_{i,0}$ instead of a parameter θ_i.
However, the initial values are not normalized and are sometimes very close to 0. To prevent numerical issues from distorting the results, we do also accept absolute CI widths < 0.01 and absolute estimation errors < 0.005 as practically observable.

Additionally, during the observability analysis of the standard model a new phenomenon appeared. If the azimuth angle is not measured, its profile likelihood seems to stay at the minimum over all $\varphi_{\mathrm{g},0}$, i. e. indicating structural unobservability. However, looking at the χ^2_{PL} course in detail, one can see that it actually varies over different values of $\varphi_{\mathrm{g},0}$ in form of a cosine, as is exemplary shown in Figure 4.6. An exploration of the standard model equations reveals that the azimuth angle does not occur by itself, but only as $\cos\psi_b$. Furthermore, the cosine of the PL curve has a time period of $\approx \frac{2\pi}{3}$, which can be found in the angle equation,

$$\psi_b = \varphi_g + \Delta\varphi + 2\pi/3\,(b-1)\,,$$

and is due to the $b_{\max} = 3$ blades of the wind turbine.

Even if its PL curve is not flat, we consider the azimuth angle to be structurally unobservable, since it never reaches the threshold but varies weekly around a constant mean value.
With the interpretation of both identifiability concepts defined, the results for both models are now presented.

First, the basic model and its three states are analyzed. The chosen scenarios are the same as in the previous section. However, the statements in the table now do not refer to the parameters given in every row, but to the initial values of the fore-aft tower velocity, the generator angular speed and the fore-aft tower position, i. e.

$$\boldsymbol{x}_0 = \begin{bmatrix} \dot{x}_{\mathrm{T},0} & \dot{\varphi}_{\mathrm{g},0} & x_{\mathrm{T},0} \end{bmatrix}^{\mathrm{T}},$$

in this order.

Table 4.12.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Basic_3Z`, wind scenario: `high_turb`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X	+s+p +s+p +s+p	+s−p +s+p +s−p	+s+p +s+p +s+p
AirDensity	+s+p +s+p +s+p	+s−p +s+p +s−p	+s+p +s+p +s+p
AirDensity+Omega0X	+s+p +s+p +s+p	+s−p +s+p +s−p	+s+p +s+p +s+p

Table 4.13.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X	+s+p +s+p +s+p	+s−p +s+p +s−p	+s∗p +s∗p +s∗p
AirDensity	+s∗p +s+p +s+p	+s−p +s+p +s−p	+s∗p +s∗p +s∗p
AirDensity+Omega0X	+s∗p +s+p +s+p	+s−p +s+p +s−p	+s∗p +s∗p +s∗p

Tables 4.12 and 4.13 represent the observability of the states with the wind considered as a known input. First, it can be seen that all states are structurally observable for every constellation (which is also valid if the wind is identified). If both generator speed and tower acceleration are measured, they are also practically observable in any case, except when data from the advanced model is used and the air density is identified. Then, the tower velocity is only partly practically observable, i. e. the relative estimation error is slightly above 10 %.

If only the generator speed is measured, the tower speed and position are practically unobservable, no matter which parameters are identified. At the same time the generator angular speed is always practically observable, which is obvious since it is directly proportional to the

generator speed in rpm and therefore known.
On the other hand, measuring only the tower acceleration is enough to practically observe all states, at least when data from the basic model itself is used. If data from the advanced model is used, all states are still partly practically observable for every parameter constellation.

Table 4.14.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Basic_3Z`, wind scenario: `steady`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X+Wind1VelX	++ ++ ++	+− ++ +−	++ ++ +∗
AirDensity+Wind1VelX	++ ++ ++	+− ++ +−	++ +∗ +∗
AirDensity+Omega0X+Wind1VelX	++ ++ ++	+− ++ +−	++ +∗ +∗

Table 4.15.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `steady`

Measured: / Identified:	GenSpeed+NcIMUTAxs	GenSpeed	NcIMUTAxs
Omega0X+Wind1VelX	++ ++ ++	+− ++ +−	++ +∗ +∗
AirDensity+Wind1VelX	++ ++ +∗	+− ++ +−	++ +∗ +−
AirDensity+Omega0X+Wind1VelX	++ ++ +∗	+− ++ +−	++ +∗ +−

Tables 4.14 and 4.15 refer to the scenarios with the wind velocity being identified as a constant parameter. Here, the practical observability is generally worse.
Only if both outputs are measured and data from the basic model is used are all three states practically observable with all three parameters identified, which then is also valid for every parameter constellation.
However, if data from the advanced model is used, the tower position is only partly practically observable when the air density and wind velocity are identified together, both with and without the eigenfrequency being identified, even if all outputs are measured. It is noteworthy that in comparison to the scenario with the wind being known, it is the tower *position* instead of the tower velocity which is only partly practical identifiable.

If only the generator speed is measured, the results are the same as if the wind velocity is not identified. Only the generator angular speed is practically observable, tower velocity and position are practically unobservable.

If only the tower acceleration is measured, at least the tower velocity is always practically observable. With data from the basic model, the tower position is only partly practically observable

and becomes practically unobservable when the air density is identified with data from the advanced model. The generator angular speed is generally partly practically observable, except if data from the basic model is used and only the eigenfrequency is identified together with the wind. Then, it is practically observable.

Standard Model

Next, the standard models qualitative observability is analyzed for the same constellations as previous identifiability section. Again, the statements given in the table do not refer to the identified parameters, but to the initial values of the eight states. Those are the tower speed both in x- (fore-aft) and in y-direction (side-side), the drive-train angular speed, the generator angular speed, the drive-train torsion, the generator azumith angle and the tower position in both x- and y-direction, i. e.

$$\boldsymbol{x}_0 = \begin{bmatrix} \dot{x}_{\mathrm{T},0} & \dot{y}_{\mathrm{T},0} & \dot{\varphi}_{\mathrm{g},0} & \Delta\dot{\varphi}_0 & x_{\mathrm{T},0} & y_{\mathrm{T},0} & \varphi_{\mathrm{g},0} & \Delta\varphi_0 \end{bmatrix}^{\mathrm{T}}.$$

Since we analyzed a total of 120 constellations for each of the 8 states, we refrain to point out the most important and interesting results. The complete results can can be found in Tables 4.16 - 4.19.

Again, we first consider the constellations with the wind being a known input given in Tables 4.16 and 4.17. If all outputs are measured, i. e. the generator speed, the azimuth angle and both fore-aft and side-side tower accelerations, the results are very similar for both data sets from the standard model itself as well as from the advanced model. All states are at least structurally observable. The fore-aft tower velocity is practically observable for every constellation, but the side-side velocity is only partly practically observable, which probably is due to bigger influence of the fore-aft velocity and its correlation with the wind velocity. As a result, the fore-aft position is also better observable than the side-side position.
The drive-train angular speed is generally practically unobservable according to our criteria.
Otherwise, the side-side velocity and position are the only states that are only partly practically observable, even if all outputs are measured. For every other measurement constellation, i. e. if the side-side acceleration is not measured, they even become structurally unobservable. (With data from the advanced model and only the air density being identified, the fore-aft position is only partly practically observable, too. However, that is due to an estimation error of 10.1 %, i. e. it is nearly practically observable according to our criteria.)

If the generator speed is measured together with the fore-aft tower acceleration, the absolute azimuth angle becomes structurally unobservable, which is true for every constellation where it is not measured directly and due to it not appearing in any equation directly. (This issue has been covered before, see Figure 4.6.)

If the generator speed is measured together with the azimuth angle, no tower velocity or position is practically observable. Obviously, the azimuth angle itself becomes observable.

Table 4.16.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified. Order of the states: $\left[\dot{x}_T \quad \dot{y}_T \quad \dot{\varphi}_g \quad \Delta\dot{\varphi} \quad x_T \quad y_T \quad \varphi_g \quad \Delta\varphi\right]^T$
Model: `Standard_8Z`, data source: `Standard_8Z`, wind scenario: `high_turb`

Measured: / Identified:	GenSpeed+Azimuth+NcIMUTAxs+NcIMUTAys
Omega0X	+s+p +s∗p +s+p +s−p +s+p +s∗p +s+p +s+p
AirDensity	+s+p +s∗p +s+p +s−p +s+p +s∗p +s+p +s+p
Omega0X+AirDensity	+s+p +s∗p +s+p +s−p +s+p +s∗p +s+p +s+p

	GenSpeed+NcIMUTAxs
Omega0X	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
Omega0X+AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p

	GenSpeed+Azimuth
Omega0X	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p
AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p
Omega0X+AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p

	GenSpeed
Omega0X	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
Omega0X+AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p

	NcIMUTAxs
Omega0X	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s−p
AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s−p
Omega0X+AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s−p

If only the generator speed is measured, the biggest difference is that the azimuth angle becomes unobservable. However, for some reason the fore-aft tower position becomes practically observable if only the air density is identified, which is valid for both data sources.

If only the fore-aft tower acceleration is measured, only the fore-aft tower velocity, position and the azimuth angular speed are practically observable. However, when using data from the advanced model, the azimuth angular speed becomes practically unobservable if the eigenfrequency is identified.

Table 4.17.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified. Order of the states: $\left[\dot{x}_T \quad \dot{y}_T \quad \dot{\varphi}_g \quad \Delta\dot{\varphi} \quad x_T \quad y_T \quad \varphi_g \quad \Delta\varphi\right]^T$
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`

Measured: / Identified:	GenSpeed+Azimuth+NcIMUTAxs+NcIMUTAys
Omega0X	+s+p +s∗p +s+p +s−p +s+p +s∗p +s+p +s+p
AirDensity	+s+p +s∗p +s+p +s−p +s∗p +s∗p +s+p +s+p
Omega0X+AirDensity	+s+p +s∗p +s+p +s−p +s+p +s∗p +s+p +s+p

	GenSpeed+NcIMUTAxs
Omega0X	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
Omega0X+AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p

	GenSpeed+Azimuth
Omega0X	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p
AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p
Omega0X+AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p

	GenSpeed
Omega0X	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
AirDensity	+s−p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
Omega0X+AirDensity	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p

	NcIMUTAxs
Omega0X	+s+p −s−p +s−p +s−p +s+p −s−p −s−p +s−p
AirDensity	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s−p
Omega0X+AirDensity	+s+p −s−p +s−p +s−p +s+p −s−p −s−p +s−p

Tables 4.18 and 4.19 refer to the constellations with the wind being identified as a constant parameter. However, the comparison of these two does not yield much new information. There are mainly some constellations for which some states become only partly practically observable if data from the advanced model is used instead of from the standard model itself. Additionally, if all outputs are measured and only the eigenfrequency is identified together with the wind velocity, the fore-aft tower position is only partly practically observable with data from the standard model, but fully practically observable with data from the advanced model. However,

this must be due to an unlucky chosen data set, since it is practically observable also for data from the standard model if the air density is identified additionally.

Since there also have been no big differences between the two data sources with the wind velocity as a known input, we further focus on a comparison between the two input constellations and data from the standard model itself, i. e. Tables 4.16 and 4.18. The first change to notice is that the side-side tower velocity and position become fully practically observable instead of only partly if all outputs are measured. However, both initial values are very close to zero. Therefore, in both cases the relative error is bigger than 10 %. But, only if the wind is identified at the same time the absolute errors are below our chosen threshold of 0.005. For example, the absolute error of the initial side-side position is 0.0016 when the wind is identified, but with 0.0071 slightly too high if not.
Another difference is that, if only the fore-aft tower acceleration is measured, the fore-aft tower position becomes practically unobservable with the wind being identified. (If only the eigenfrequency is identified with the wind, it is at least still partly practically observable.)

However, all in all, identifying the wind velocity as an additional parameter does not lead to a significant decrease in observability. Especially if all outputs are measured, the observability seem to even increase.

Concluding, both the basic and the standard model are generally observable if all outputs are measured - even with the wind being identified. The drive-train angular speed in the standard model is the only state which stays practically unobservable regardless of the constellation. However, that is due to our rigid interpretation of practical observability, since limited confidence intervals can be found. Therefore, it might still be a possible to observe all of the states, e. g. for the use of a state space controller.

Table 4.18.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified. Order of the states: $\left[\dot{x}_\mathrm{T} \quad \dot{y}_\mathrm{T} \quad \dot{\varphi}_\mathrm{g} \quad \Delta\dot{\varphi} \quad x_\mathrm{T} \quad y_\mathrm{T} \quad \varphi_\mathrm{g} \quad \Delta\varphi\right]^\mathrm{T}$
Model: Standard_8Z, data source: Standard_8Z, wind scenario: steady

Measured: / Identified:	GenSpeed+Azimuth+NcIMUTAxs+NcIMUTAys
Omega0X+Wind1VelX	+s+p +s+p +s+p +s−p +s∗p +s+p +s+p +s+p
AirDensity+Wind1VelX	+s+p +s+p +s+p +s−p +s+p +s+p +s+p +s+p
AirDensity+Omega0X+Wind1VelX	+s+p +s+p +s+p +s−p +s+p +s+p +s+p +s+p

	GenSpeed+NcIMUTAxs
Omega0X+Wind1VelX	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
AirDensity+Wind1VelX	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
AirDensity+Omega0X+Wind1VelX	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p

	GenSpeed+Azimuth
Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p
AirDensity+Wind1VelX	+s−p −s−p +s+p +s−p +s−p +s−p +s+p +s+p
AirDensity+Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p

	GenSpeed
Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
AirDensity+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
AirDensity+Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p

	NcIMUTAxs
Omega0X+Wind1VelX	+s+p −s−p +s+p +s−p +s∗p −s−p −s−p +s+p
AirDensity+Wind1VelX	+s+p −s−p +s−p +s−p +s−p −s−p −s−p +s+p
AirDensity+Omega0X+Wind1VelX	+s+p −s−p +s−p +s−p +s−p −s−p −s−p +s−p

Table 4.19.: Overview of the structural and practical observability, in dependence of the measured outputs and the parameters to be identified. Order of the states: $\left[\dot{x}_T \quad \dot{y}_T \quad \dot{\varphi}_g \quad \Delta\dot{\varphi} \quad x_T \quad y_T \quad \varphi_g \quad \Delta\varphi\right]^T$
Model: Standard_8Z, data source: Advanced_21Z, wind scenario: steady

Measured: / Identified:	GenSpeed+Azimuth+NcIMUTAxs+NcIMUTAys
Omega0X+Wind1VelX	+s+p +s+p +s+p +s−p +s+p +s+p +s+p +s+p
AirDensity+Wind1VelX	+s+p +s+p +s+p +s−p +s∗p +s+p +s+p +s+p
AirDensity+Omega0X+Wind1VelX	+s+p +s+p +s+p +s−p +s∗p +s+p +s+p +s+p

	GenSpeed+NcIMUTAxs
Omega0X+Wind1VelX	+s+p −s−p +s+p +s−p +s+p −s−p −s−p +s+p
AirDensity+Wind1VelX	+s+p −s−p +s+p +s−p +s∗p −s−p −s−p +s+p
AirDensity+Omega0X+Wind1VelX	+s+p −s−p +s+p +s−p +s∗p −s−p −s−p +s+p

	GenSpeed+Azimuth
Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p
AirDensity+Wind1VelX	+s−p +s−p +s+p +s−p +s−p +s−p +s+p +s+p
AirDensity+Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p +s+p +s+p

	GenSpeed
Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
AirDensity+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p
AirDensity+Omega0X+Wind1VelX	+s−p −s−p +s+p +s−p +s−p −s−p −s−p +s+p

	NcIMUTAxs
Omega0X+Wind1VelX	+s+p −s−p +s−p +s−p +s∗p −s−p −s−p +s−p
AirDensity+Wind1VelX	+s+p −s−p +s−p +s−p +s−p −s−p −s−p +s−p
AirDensity+Omega0X+Wind1VelX	+s+p −s−p +s−p +s−p +s−p −s−p −s−p +s−p

4.3 Identifiability Analysis

In this section, we refrain to a selection of relevant scenarios in practice and analyze their identifiability in Detail. Therefore, three scenarios are chosen. First, the air density and the eigenfrequency are identified, i. e. the wind is treated as an input. Secondly, the eigenfrequency is assumed to be known but the wind velocity is to estimated together with the air density instead. Thirdly, all three air density, eigenfrequency and wind velocity are identified at the same time.

For all simulations in this section, the same settings have been used as before, i. e. as given in Table 4.2. Only the `data_scenario` has been changed to `'steady'` if the wind velocity is estimated and all observables are measured for every constellation shown here.

4.3.1 Identifying Air Density + Eigenfrequency

First, we take a look at the identfiability of both air density and eigenfrequency with the basic model. We already know from Tables 4.4 and 4.5 that they both are structurally and practically identifiable, no matter if data from the basic model itself or from the advanced model is used. Figure 4.7 shows the parameters profile likelihood progressions for both cases.

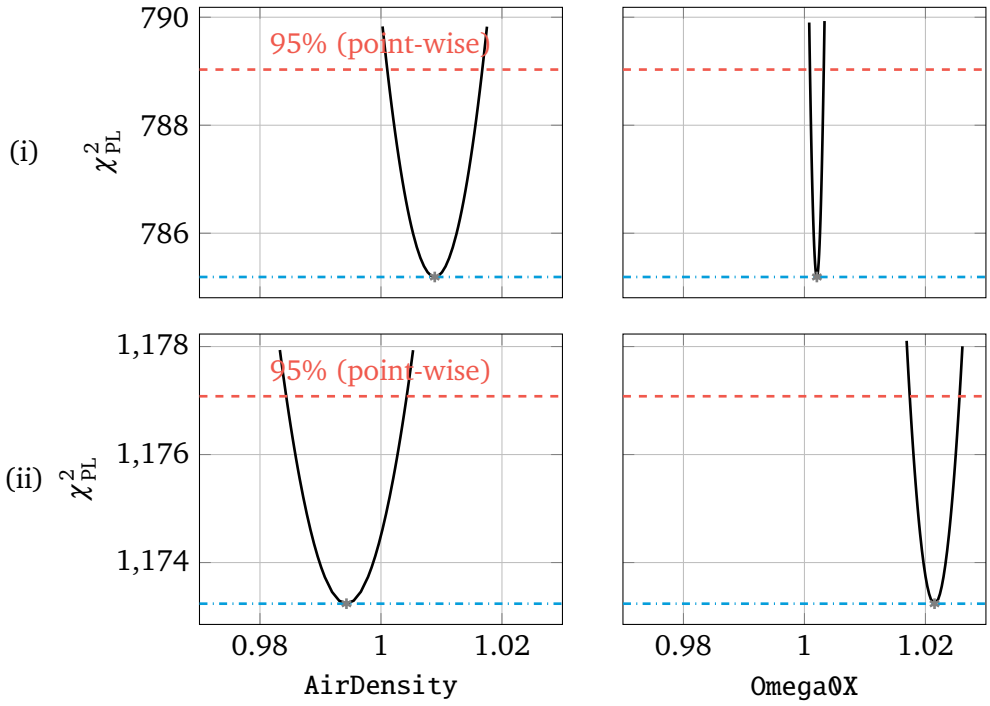

Figure 4.7.: Profile likelihood progressions for both air density and eigenfrequency.
Model: `Basic_3Z`, data sources: `Basic_3Z` (i) and `Advanced_21Z` (ii)

As can be seen, the absolute χ^2_{PL} value is much higher if data from the advanced model is used (ii) than if from the basic model itself (i), which corresponds to a worse fit of the data points.

However, this does not effect the threshold $\Delta_\alpha = \chi^2(\alpha, df)$, i. e. the vertical distance between the best fit and the confidence interval (CI) limit, which is ≈ 4 here. The upper red dashed line indicates the limit for the 95 % CI, whereby the CIs are valid only point-wise, which means that the calculated limits would be only valid if all other parameter estimations were true. The lower blue dash-dotted line only indicates the χ^2_{PL} value of the best fit. If a vertical gray line is plotted, it is only to emphasize the true parameter value.

All fits are negligible close to the true (normalized) value of 1. However, only for the air density with data from the advanced model (ii) the 95 % interval contains this true value. If data from the basic model itself is used (i), the lower bounds are 0.1 % and 0.09 % too high, but the CI widths are very narrow at the same time with 0.0157 and 0.0023. The relative estimation errors are 0.89 % and 0.21 %. This is a good example that the CI limits are not necessarily true, but still an indication of the estimation accuracy.

If data from the advanced model is used (ii), the estimation errors are 0.57 % and 2.15 %, i. e. only the eigenfrequency estimation has a significant error here. However, the CI for the air density is wider than for the eigenfrequency (0.02 to 0.008). This might be an indication that the correlation between CI width and estimation error decreases with an increasing model error, which will be discussed in Section 4.5.1.

If data from the advanced model is used (ii), the true value is ≈ 1 % outside of the CI, which shows that the 95 % likelihood is not true due to the significant model errors. However, both parameters are still structural as well as practical identifiable and the estimated values are very close to the true value.

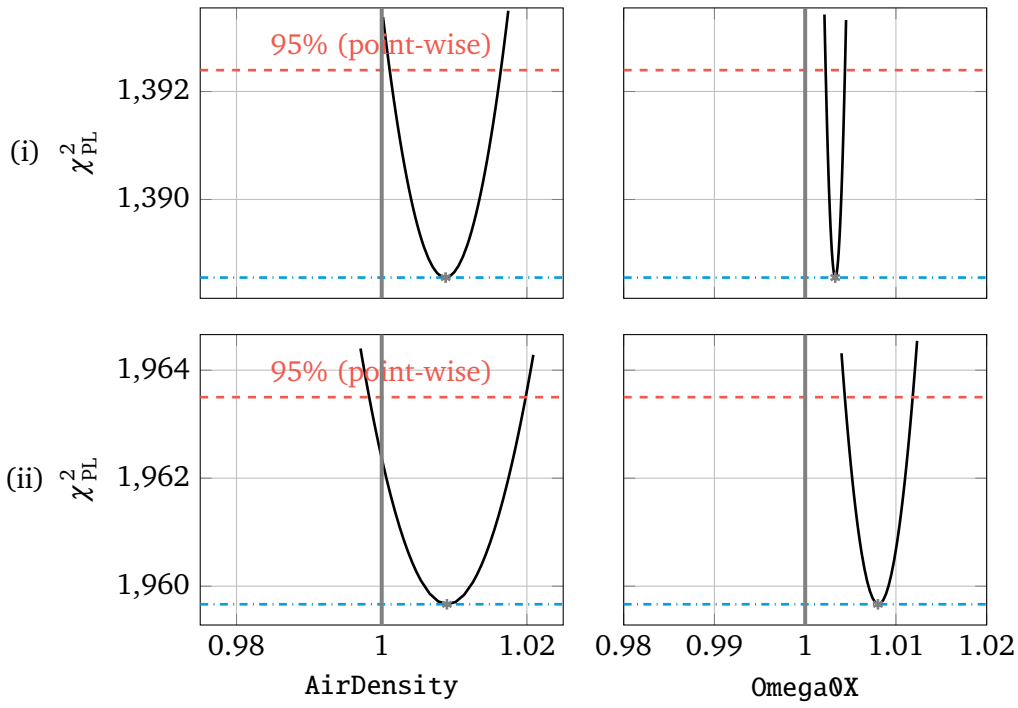

Figure 4.8.: Profile likelihood progressions for both parameters.
Model: `Standard_8Z`, data sources: `Standard_8Z` (i) and `Advanced_21Z`

Secondly, Figure 4.8 shows the results if the standard model is used, again for data from itself (i) and from the advanced model (ii). As can be seen, the differences between both data sources are smaller. The estimations are nearly the same (air density: 1.0088 (i) and 1.0090 (ii), eigenfrequency: 1.0033 (i) and 1.0080 (ii)). That seems reasonable since the standard model describes data from the advanced model better than the basic model does. The only significant difference between (i) and (ii) is that the confidence intervals are wider for (ii).

Concluding, if the wind is known, both air density and eigenfrequency can be identified very well. Only the eigenfrequency showed a significant error of 2.15 % when the basic model was used with 'real' data.

4.3.2 Identifying Air Density + Wind Velocity

Unfortunately, the wind velocity cannot necessarily be assumed known . Therefore, in this section we first assess its identifiability if the air density is estimated at the same time. Again, the basic and standard models are compared to each other and the different data sources.

Figure 4.9 shows the results for the basic model. Please note that the wind velocity is *not* normalized, but has a true value of 4. As already stated in the qualitative overview in Tables 4.6 and 4.7, only the wind velocity estimation with data from the basic model itself (i) has an relative error of $3.31\% < 10\%$ and is therefore considered practically observable. The air density's error is slightly to high with 11.47 %.
If data from the advanced model is used (ii), the errors are unacceptable high with 65.17 % and 14.71 % for the air density and wind velocity, respectively. The confidence intervals widths are wider, too.

Figure 4.10 shows the results for the standard model, which are very similar. If data from the standard model itself is used (i), the relative estimation errors are 13.36 % and 3.14 %. They again increase unacceptable high with data from the advanced model (ii), i. e. 48.46 % and 11.59 %.

In general, the air density is estimated too high and the wind velocity too low. This is true for both models and data sources. The errors for both models with data from themselves are similar. However, if data from the advanced model is used, the estimation errors decrease for the standard model, whereby they are still too big. Concluding, the estimation of the wind velocity as a parameter together with the air density does not work well enough if a simplified model is used. If the wind is unknown, one might consider to obtain information about the air density somehow else, e. g. using temperature sensors.

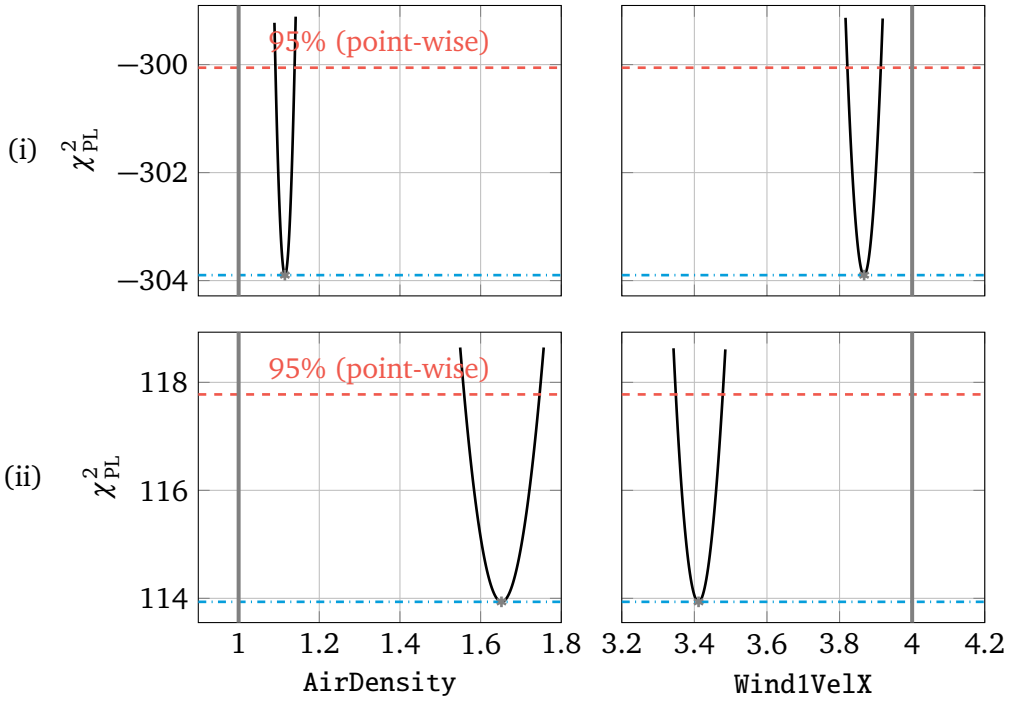

Figure 4.9.: Profile likelihood progressions for both parameters.
Model: `Basic_3Z`, data sources: `Basic_3Z` (i) and `Advanced_21Z` (ii)

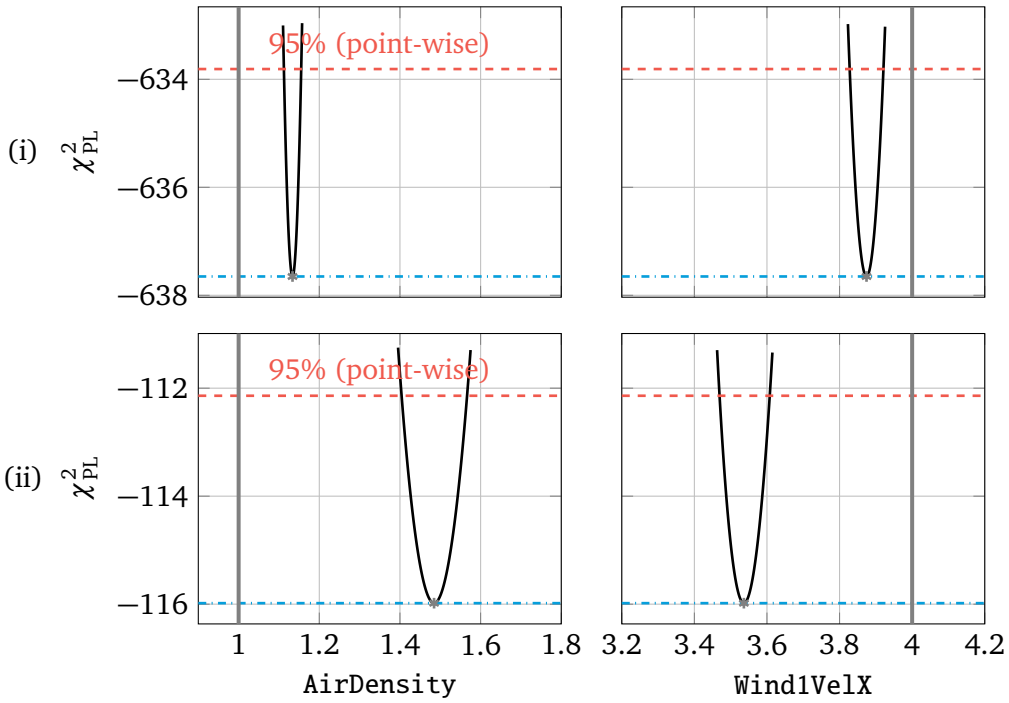

Figure 4.10.: Profile likelihood progressions for both parameters.
Model: `Standard_8Z`, data sources: `Standard_8Z` (i) and `Advanced_21Z` (ii)

4.3.3 Identifying Eigenfrequency + Wind Velocity

Since in the previous section we showed that identifying the wind velocity together with the air density does not work well, we investigate its identifiability together with the eigenfrequency instead. Otherwise, the same constellations are analyzed as above.

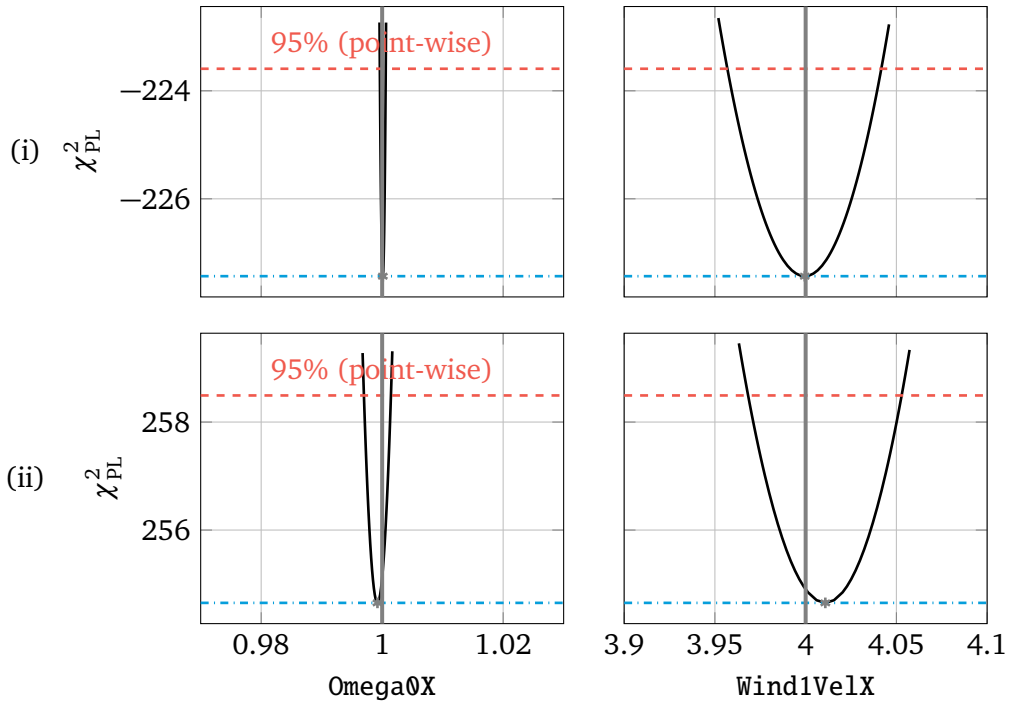

Figure 4.11.: Profile likelihood progressions for both parameters.
Model: `Basic_3Z`, data sources: `Basic_3Z` (i) and `Advanced_21Z` (ii)

Figure 4.11 shows the results for the basic model. In contrast to the air density, the eigenfrequency is very well identifiable together with the wind velocity. If data from the basic model itself is used (i), both relative errors are negligible small and $< 0.012\,\%$. Even for data from the advanced model (ii), the relative errors are very small with $0.08\,\%$ and $0.27\,\%$.

Figure 4.12 shows the results for the standard model. As can be seen, the estimations are still good with with errors of $2.22\,\%$ and $1.00\,\%$ for data from the standard model itself (i) and $0.56\,\%$ and $0.63\,\%$ for data from the advanced model (ii). However, the eigenfrequency seems to be worse identifiable for the standard model than for the basic model. The results from Section 4.3.1 (Identifying Air Density + Eigenfrequency) support this conclusion, since the estimation errors for the eigenfrequency with the standard model were higher, too. Respecting both system's equations, one can see that the eigenfrequency affects 2/3 of the states for the basic model but only half of the states for the standard model, which might be an explanation. Furthermore, Figure 4.12 highlights that, in contrast to the air density, the eigenfrequency tends to be estimated too low, whereby the wind velocity rather is estimated too high.

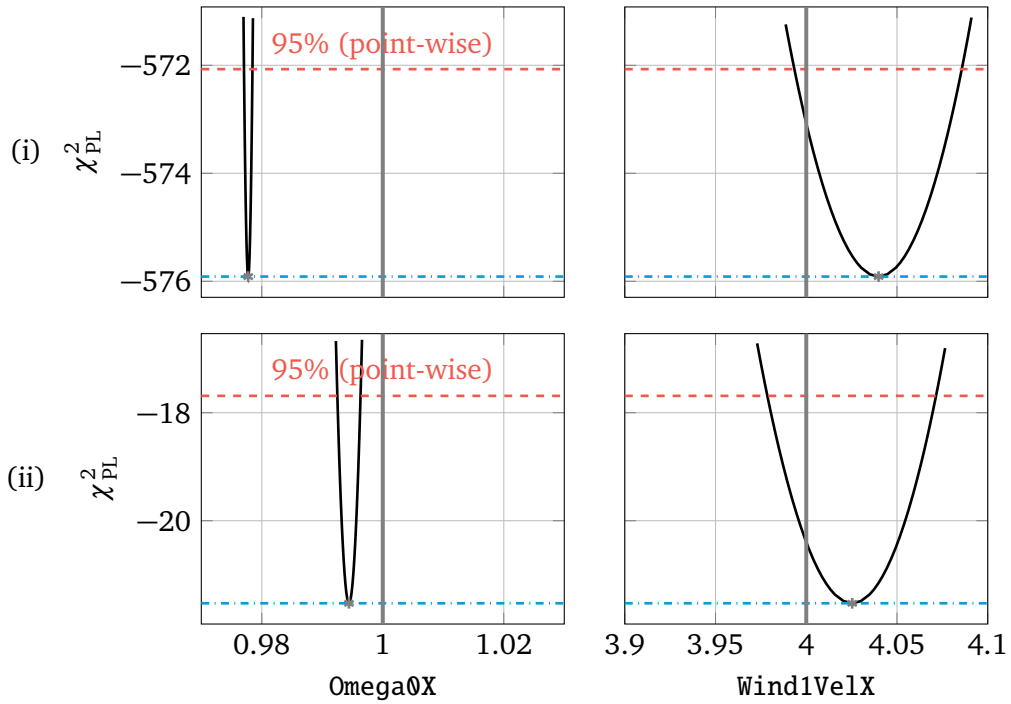

Figure 4.12.: Profile likelihood progressions for both parameters.
Model: `Standard_8Z`, data sources: `Standard_8Z` (i) and `Advanced_21Z` (ii)

Concluding, the eigenfrequency can be identified together with the wind velocity for both models and regardless of the data source. This is an important result, since there is no easy alternative to determine its progress over the time. Next, we investigate whether is stays identifiable if the air density is estimated additionally.

4.3.4 Identifying Air Density + Eigenfrequency + Wind Velocity

Since in Section 4.3.2 it has been shown that identifying the air density together with the wind velocity does not work well, we cannot expect it to work with the eigenfrequency being estimated additionally. However, it is worth investigating if their as well as the eigenfrequency's identifiability further decreases.

Figure 4.13 shows the results for the basic model. As expected, the air density is estimated too high and the wind velocity to low, whereas the eigenfrequency again is estimated very accurately. Compared to Figure 4.9, there are no significant differences whether the eigenfrequency is identified additionally or not. The same is true for the eigenfrequency's identifiability, it is not affected whether the air density is estimated additionally or not.

Figure 4.14 shows the results for the standard model. Again, in comparison to Figure 4.10, no significant differences can be seen for the air density or wind velocity.
However, identifying the eigenfrequency with the standard model and data from itself led to an estimation error of 2.22 % in Figure 4.12, i. e. when it was only estimated together with the wind. Here, it is estimated extremely accurate with an estimation error of only 0.01 %. If

data from the advanced model is used, the estimation is better, too, with an error of 0.33 % (compared to 0.63 % without identifying the air density).

Concluding, the air density's and wind velocity's identifiability is neither better nor worse with or without the eigenfrequency being identified. Only the eigenfrequency's identifiability when using the standard model is improved when the air density is estimated at the same time.

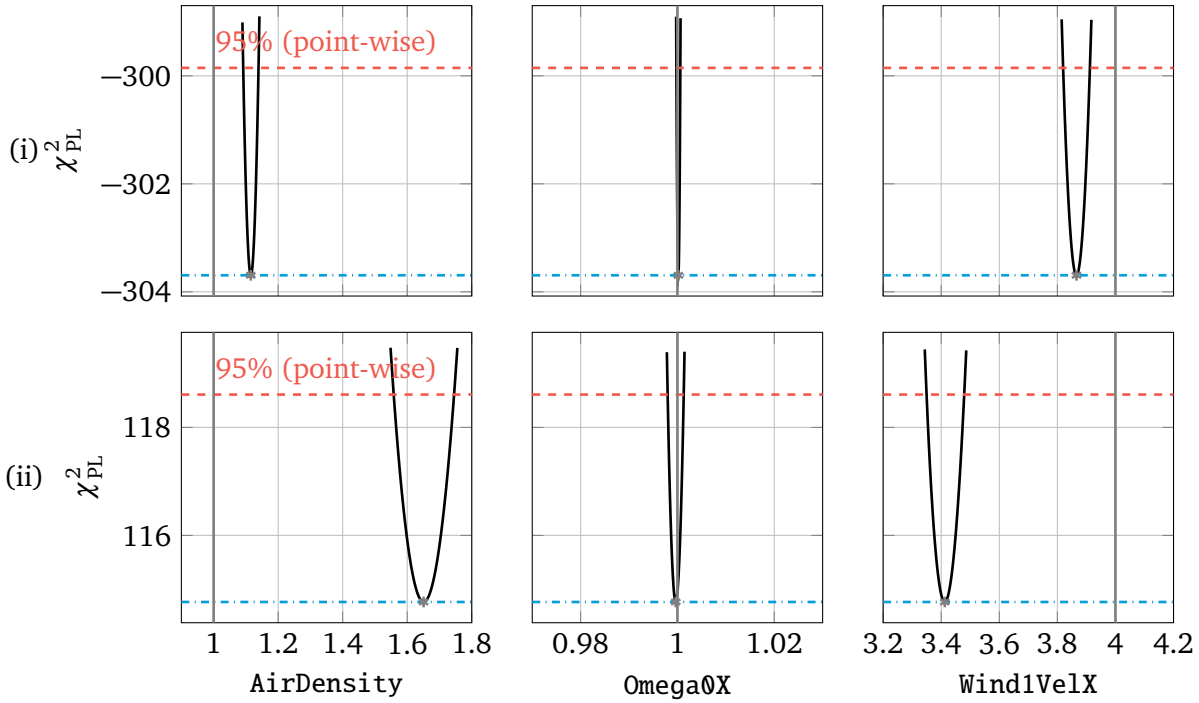

Figure 4.13.: Profile likelihood progressions for all three parameters.
Model: `Basic_3Z`, data sources: `Basic_3Z` (i) and `Advanced_21Z` (ii)

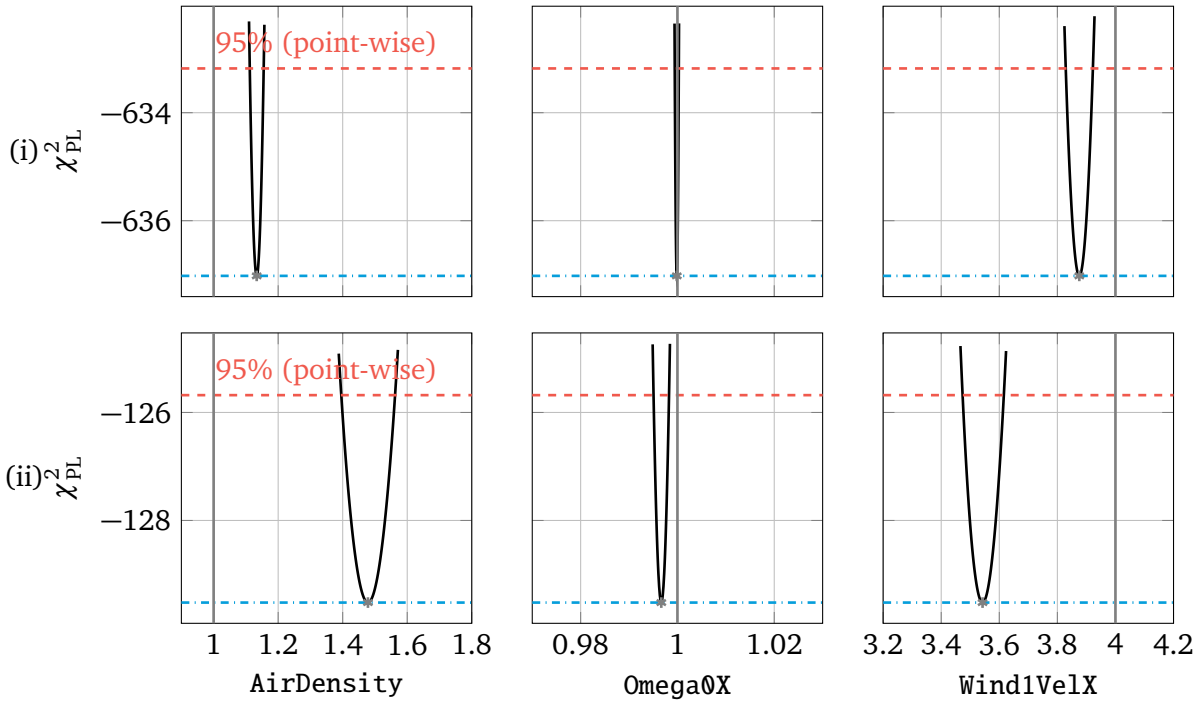

Figure 4.14.: Profile likelihood progressions for all three parameters.
Model: `Standard_8Z`, data sources: `Standard_8Z` (i) and `Advanced_21Z` (ii)

4.4 Observability Analysis

After both models' identifiability has been analyzed, we now consider their observability in detail. We do so by considering the profile likelihood progression of the system's initial values as well as calculating the prediction profile likelihood (PPL) for multiple time points. Furthermore, we plot and compare the D2D state estimations with the actual data.
In the first subsection, the wind is assumed to be a known input, whereby the air density as well as the eigenfrequency is identified. Secondly, we again consider the wind velocity to be a (constant) free parameter and estimate it together with the air density and eigenfrequency.

The general simulation settings are mostly the same as before. For convenience, they are summarized again in Table 4.20. As can be seen, we simulated both models only with measurement data from the advanced model, i. e. with 'true' data, to assess their observability in a real-world environment.

Table 4.20.: General simulation settings for the qualitative identifiability and observability assessment.

Variable	Possible values
model	'Basic_3Z' or 'Standard_8Z'
data_source	'Advanced_21Z'
data_scenario	'high_turb' or 'steady'
data_nDataPoints	200
data_nStart	1000
data_dt	0.1
free_parameters	'Omega0X+AirDensity'
free_initials	'all'
noise_snr_db	30
noise_seed	1
interpolation	@writeSuperSplines
inputs	'GenTq+Wind1VelX' or 'GenTq'
observables	'all'
doPLE	1
doPPL	1
nFitLHS	25

4.4.1 Wind known

In this section, we first analyze the basic model's observability if the wind is considered a known input. Then, the same is done for the standard model.

With both outputs measured and the wind being an input, we know from Table 4.13 that the tower velocity is only partly observable if both eigenfrequency and air density are identified, but both the generator angular speed and the tower position are practically observable.

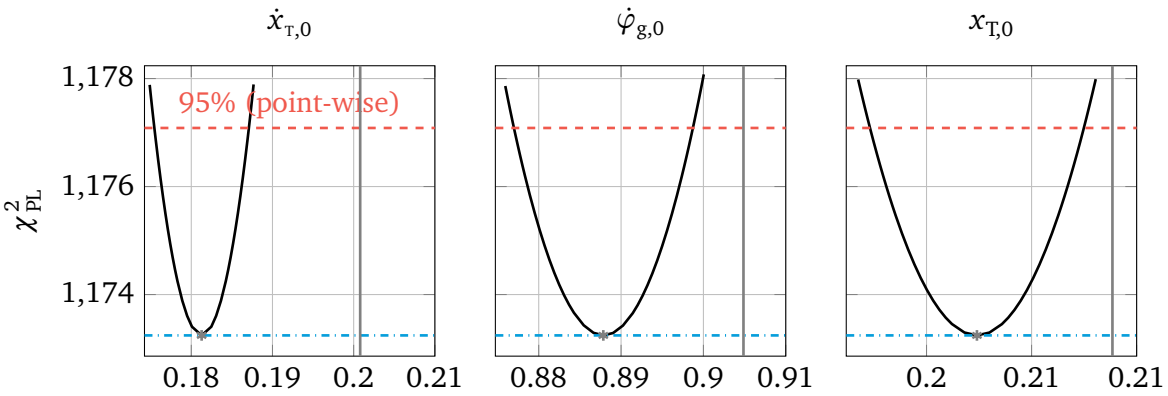

Figure 4.15.: Profile likelihood progressions for all initial state values.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`, i. e. the wind is known.

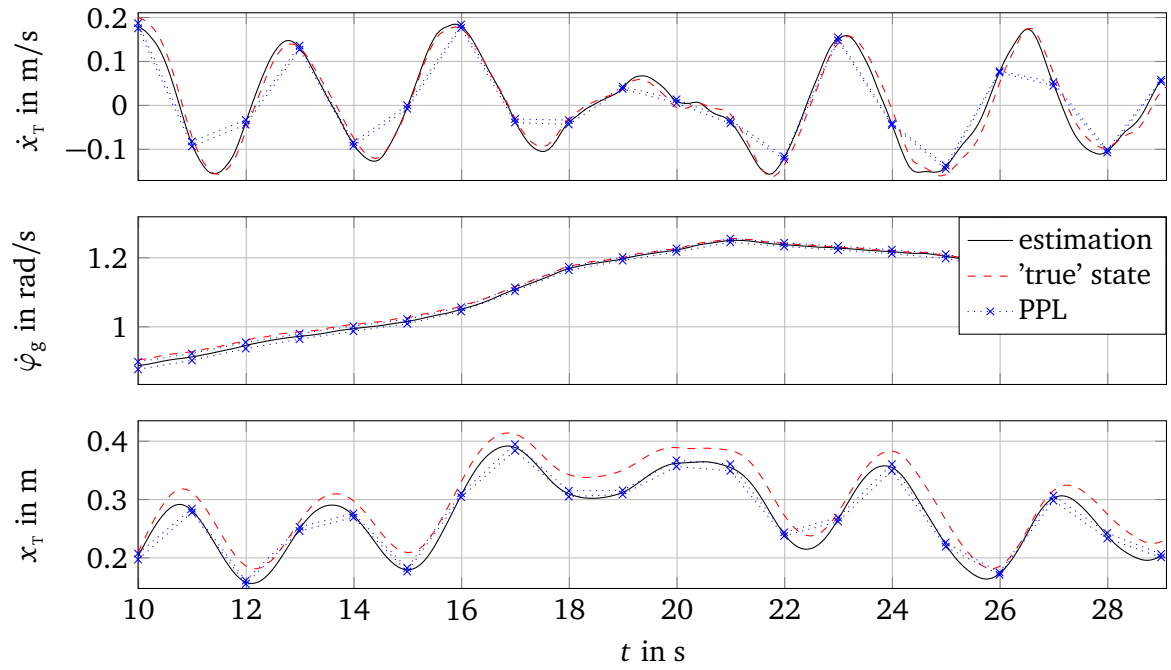

Figure 4.16.: Comparison of the 'true' states and their estimations including the PPL.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`, i. e. the wind is known.

Figure 4.15 shows the profile likelihood progressions of their initial values. As one can see, all three derived confidence intervals do not contain the true value, but only for the tower velocity the estimation error is $> 10\,\%$. However, to assess the states' observability over time, we fur-

thermore determined the prediction profile likelihood. Since its simulation is computationally very demanding, we chose to calculate the PPL values only for every second. The results for the basic model are given in Figure 4.16. Thereby, the black line represents the state estimation, the dashed red line the actual state, i. e. from the advanced model, and the blue crosses the upper and lower bound for every calculated PPL confidence interval.

As one can see, the estimated state values are generally very close to the 'true' states. Although the generator speed (rpm) is measured, the generator angular speed has a very small error, which is most likely due to modeling errors. The tower velocity is estimated very well. Its estimation error is mostly below the 10 % from the initial value. The tower position, however, is estimated less well over the time course than as for the initial value. Its estimation is always too low.

For all three states, the PPL intervals mostly do not contain the 'true' values, but are very narrow around the estimated state progression. That means that errors from uncertainty in measurements are very small. Instead, the modeling error predominates and is responsible for the deviations between the PPL confidence intervals and the actual state values.

Standard model

Next, the standard model's observability is considered for the same scenario. Figure 4.21 shows the profile likelihood progression for all states' initial values. In accordance with Table 4.17, the drive-train angular speed is practically unobservable due to a too-wide confidence interval and the side-side tower velocity and position are only partly practically observable due to estimation errors $> 10\,\%$.

Figure 4.22 shows the PPL for all eight states and gives more insight into their observability. The fore-aft tower velocity and position (plots 1 and 7) estimations are mostly good, but do contain errors if the state's direction is changed abruptly.
The estimation of the side-side tower velocity and position (plots 2 and 8) do in principle follow the true states, but contain the biggest errors, which is in compliance to their initial values being only partly identifiable. The position is generally estimated too high.
The drive-train angular velocity (plot 3) was considered practically unobservable due to the large confidence interval for its initial value, which is verified by the PPL confidence interval at $t_0 = 10\,\mathrm{s}$. Furthermore, the state's estimation oscillates multiple times stronger around 0 than the true state, which is the reason for the large PPL confidence intervals. However, that is only true for the beginning of the chosen scenario, i. e. for the transient behavior. After $t = 18\,\mathrm{s}$, the estimation is so good that the D2D framework cannot derive any PPL confidence intervals due to a too small error. For the drive-train tension (plot 5), it is very similar. Again, for the transient behavior in the beginning the estimation oscillates much more strongly than the true state. Hence, the PPL confidence intervals are very large. (The profile likelihood confidence interval of the initial value has not been considered too large since it is below our chosen numerical

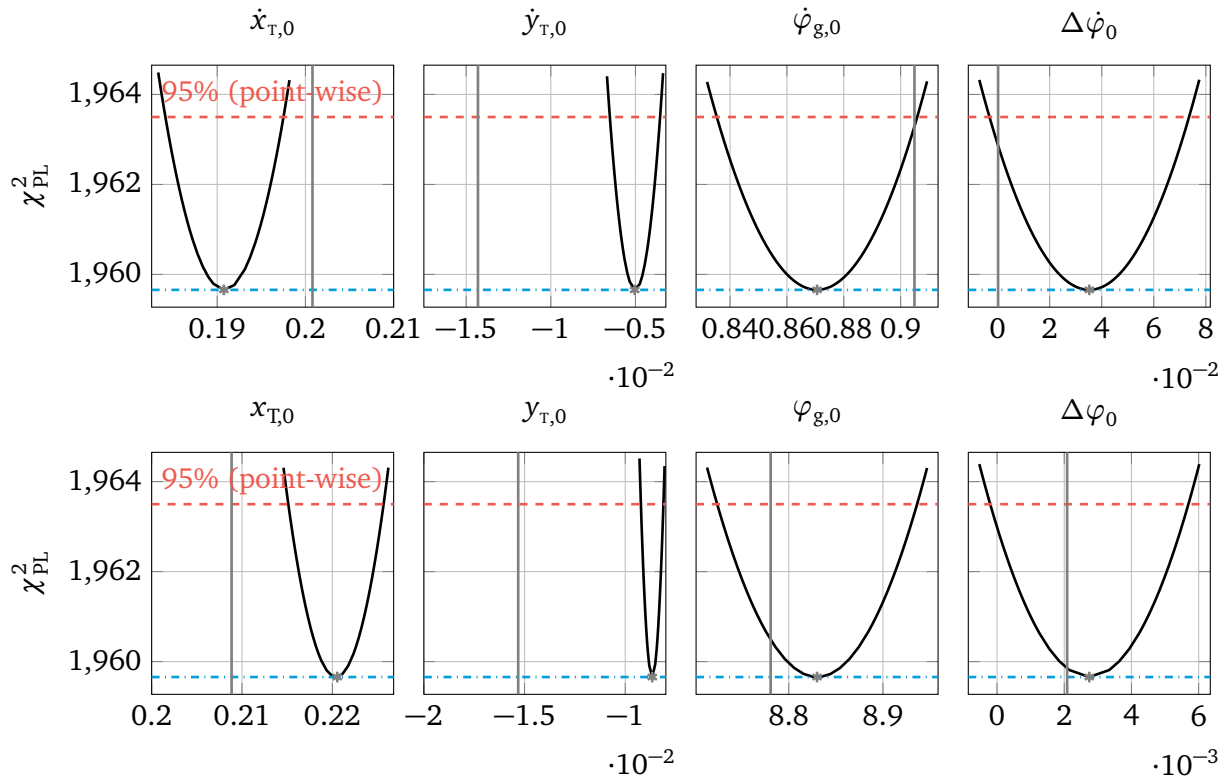

Figure 4.17.: Profile likelihood progressions for all initial state values.
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`, i. e. the wind is known.

threshold of 0.01, otherwise the state had been judged practically unobservable.) However, after the transition period, the estimation is so accurate that again no PPL confidence intervals could be derived.

For the generator azimuth angular speed (plot 4), the estimation again oscillates more strongly than the true state. However, the oscillation's amplitude is relatively small compared to the absolute value. Hence, the deviations in the transition period are smaller than for the drive-train states. In return, after the transient phase there is still a big enough error to derive confidence intervals of the PPL. In contrast, the azimuth angle (plot 4) is measured directly and is therefore 'estimated' perfectly. Therefore, no PPL intervals can be derived at any point.

Concluding, with the wind being a known input, both models are in general observable in a real-world environment, even when the air density and eigenfrequency are identified at the same time. For the basic model, only the fore-aft tower position shows significant but acceptable deviations. For the standard model, the estimated drive-train states as well as the generator angular speed all oscillate too strong during the transient behavior, but are very accurate otherwise. Only the side-side tower velocity and position show significant deviations over the whole time course, but should still be observable well enough for a real application due to their small magnitude and relevance.

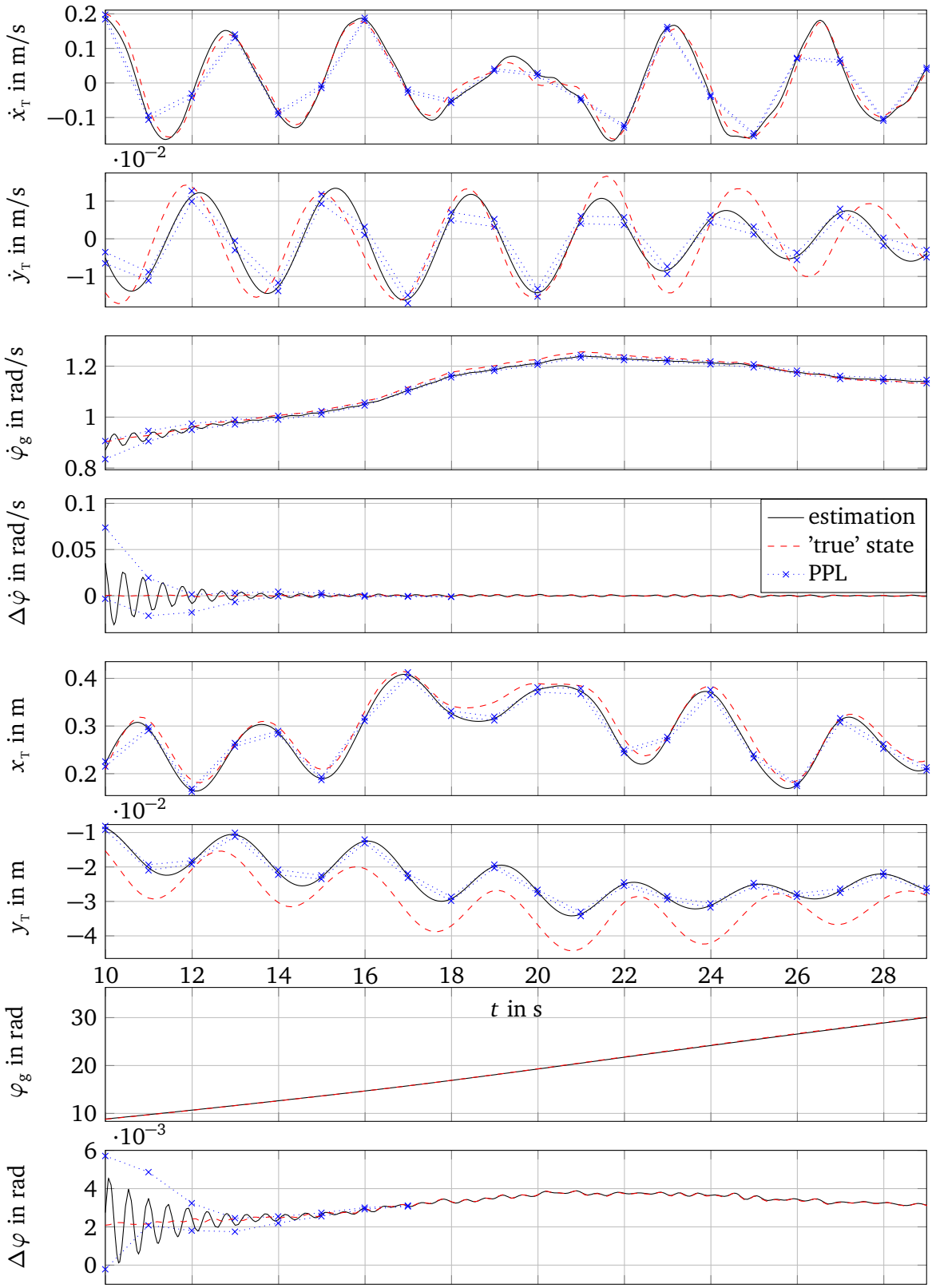

Figure 4.18.: Comparison of the 'true' states and their estimations including the PPL.
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`, i. e. the wind is known.

4.4.2 Wind unknown

As before, in this section both the basic and the standard model's observability is analyzed, but this time with the wind being identified as a constant parameter.

Basic model

First, the basic model is considered. Figure 4.19 shows the profile likelihood for the states' initial values. In contrast to when the wind is known, here, the tower position's (instead of the velocity's) initial value is considered only partly practically identifiable. For the tower velocity and the angular speed, both confidence intervals contain the true value and are reasonably wide.

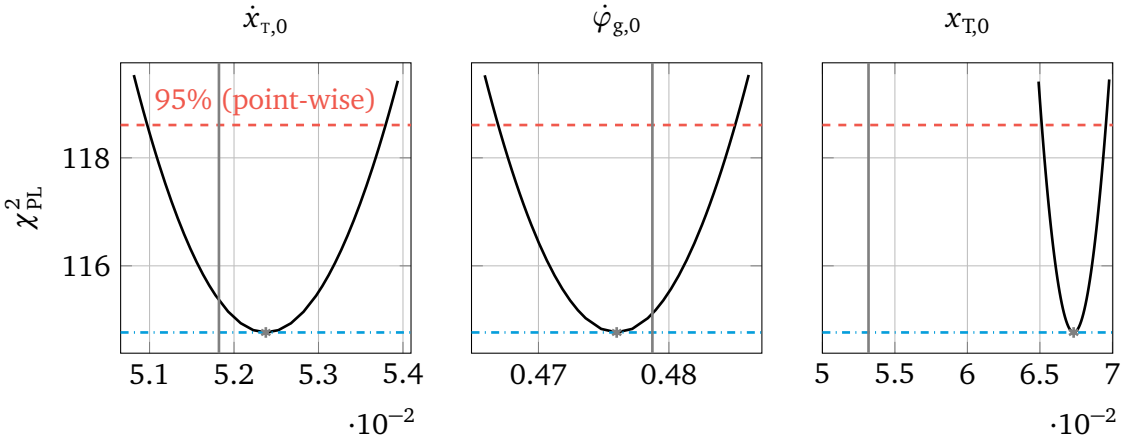

Figure 4.19.: Profile likelihood progressions for all initial state values.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `steady`, i. e. the wind is unknown.

Figure 4.20 shows the PPL for the three states. The states' estimations are in accordance with the profile likelihood of their initial values. The tower velocity and the azimuth angular speed are estimated very well. For the tower velocity, the PPL confidence intervals are extremely narrow and therefore do not necessarily contain the true value, even if the estimation error is very small.
The angular speed's estimation has a nearly constant offset and is negligible smaller than the true stated over the complete time course. However, the PPL confidence intervals are not as narrow as for the tower velocity and do always contain the true values.
As could be expected from the initial value's profile likelihood, the tower position is the least observable and has a significant error. It is estimated too high with a nearly constant offset again. The PPL confidence intervals are not as small as for the tower velocity, but do never contain the true value due to the relatively large error.

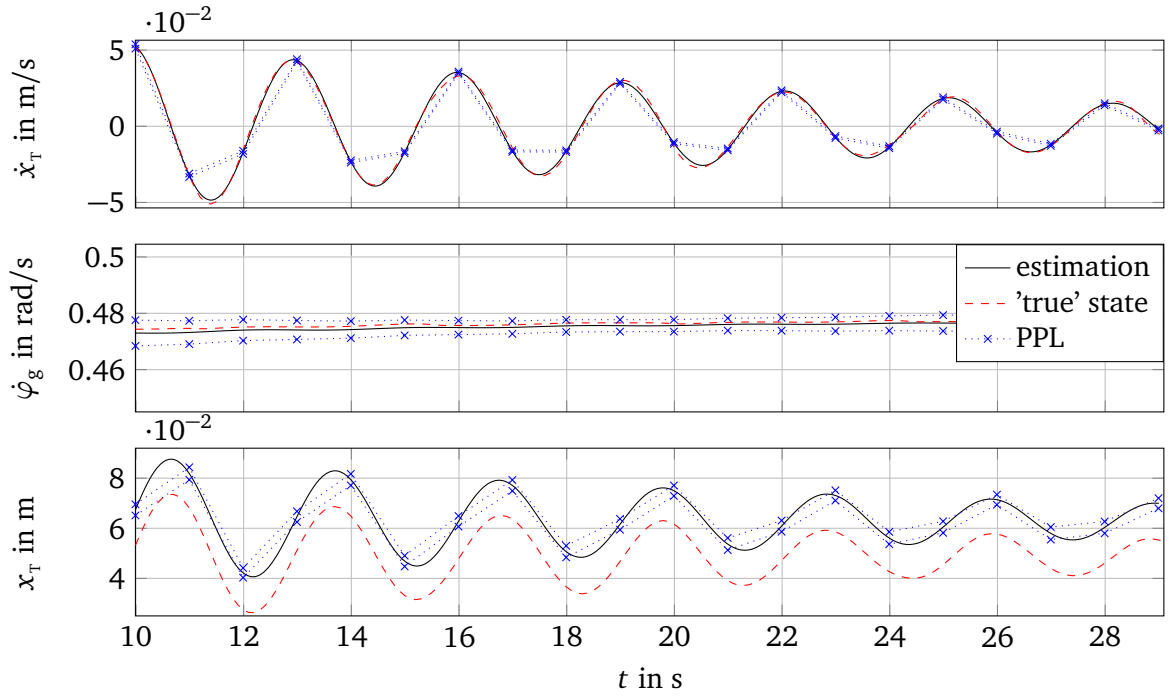

Figure 4.20.: Comparison of the 'true' states and their estimations including the PPL.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `steady`, i. e. the wind is unknown.

Standard Model

Lastly, the standard model's observability with the constant wind scenario is analyzed. Figure 4.21 shows the profile likelihood for the states' initial values. In accordance to Table 4.19, the drive-train angular speed is considered practically unobservable due to a too wide PL confidence interval of its initial value. The fore-aft tower position is only partly practically observable due to a too bad fit. The side-side tower velocity and position are considered fully practically observable since their estimation errors are below the chosen numerical threshold of 0.005. The drive-train torsion's confidence interval is considered reasonably wide since it is < 0.01.

Figure 4.22 shows the states' estimations and PPL for the chosen scenario. As can be seen, the fore-aft tower velocity (plot 1) is estimated very well and the PPL confidence intervals are extremely narrow. The fore-aft tower position (plot 7) is estimated badly not only at its initial value, but over the whole time course. It is always estimated too high by a nearly constant offset.
The side-side tower velocity (plot 2) is estimated worse than the fore-aft velocity, but still good. The oscillations' peaks are too weak sometimes, but the frequency is right and there are nearly no horizontal shifts. The side-to-side position (plot 8) is estimated worse, but still to an acceptable degree. The lower peaks are mostly estimated to weak, the higher ones sometimes too weak and sometimes too strong.
The drive-train angular speed's and tension's progressions (plots 3 and 5) are very similar to those in Figure 4.18, i. e. when the wind is known. In the transition period, they both have

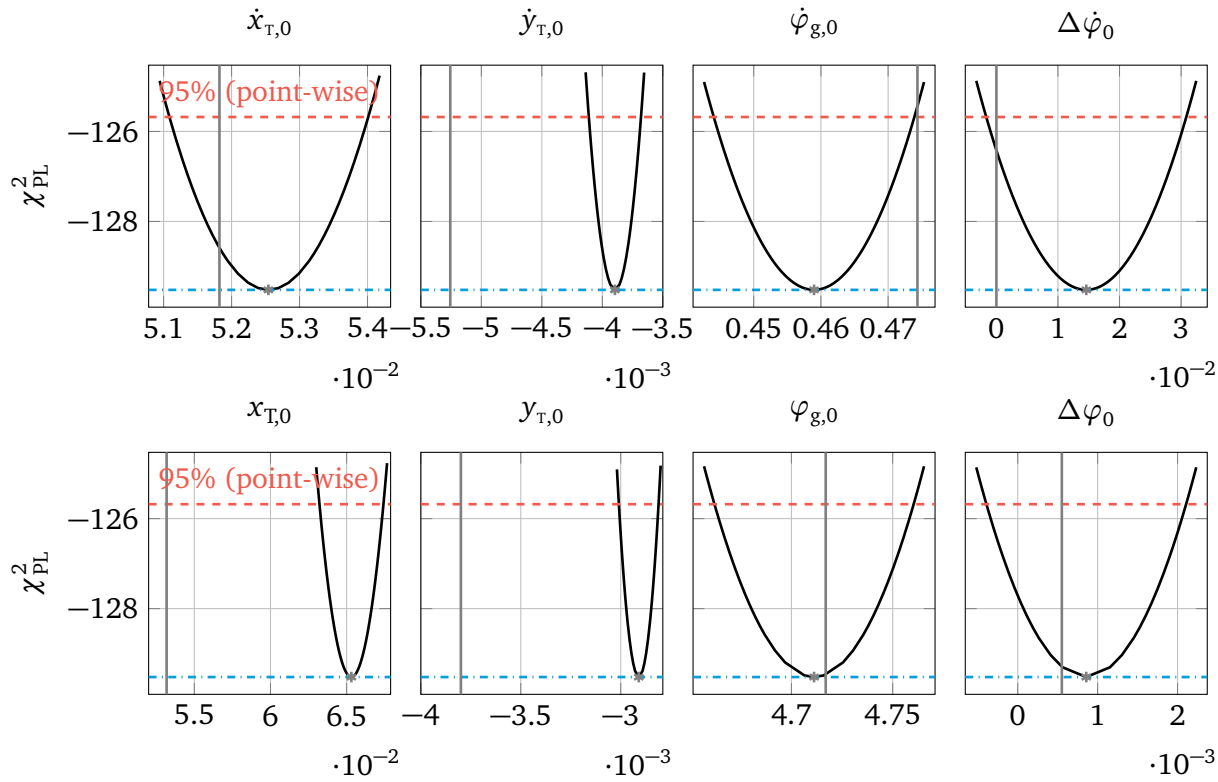

Figure 4.21.: Profile likelihood progressions for all initial state values.
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `steady`, i. e. the wind is unknown.

much stronger oscillations than the true states. Then, after $t = 18\,\mathrm{s}$, in both cases the errors are too small to determine any PPL confidence intervals.
The same applies to the generator azimuth angular speed (plot 4), which again shows additional oscillations in the transition period, too. After they fade out, the error is very small. However, the D2D framework can still derive PPL confidence intervals. The measured azimuth angle itself (plot 5) has again no deducible error and no PPL confidence intervals can be calculated.

Concluding, with the wind velocity being identified additionally to the air density and eigenfrequency, both models are still generally observable. That means that even though the wind velocity and air density cannot properly be distinguished, i. e. they are only partly practically non-identifiable, their combined effect on the model's states is estimated well enough. For the basic model, again only the fore-aft tower position shows a significant and nearly constant offset. For the standard model, the angular quantities (except the measured azimuth angle) show overly strong oscillations in the transition period, but are estimated nearly errorless afterwards. The side-side tower velocity and position are estimated even better, which may be due to the constant wind scenario. In return, the fore-aft tower position is estimated worse, but also still acceptable well.

As a different aspect, it shall be noted that the state estimations conducted by the D2D framework can be used as a benchmark test for an observer that is to be implemented for an online

application. Despite the PPL confidence intervals, the estimations by the framework evidently show how well the states can be estimated at least.

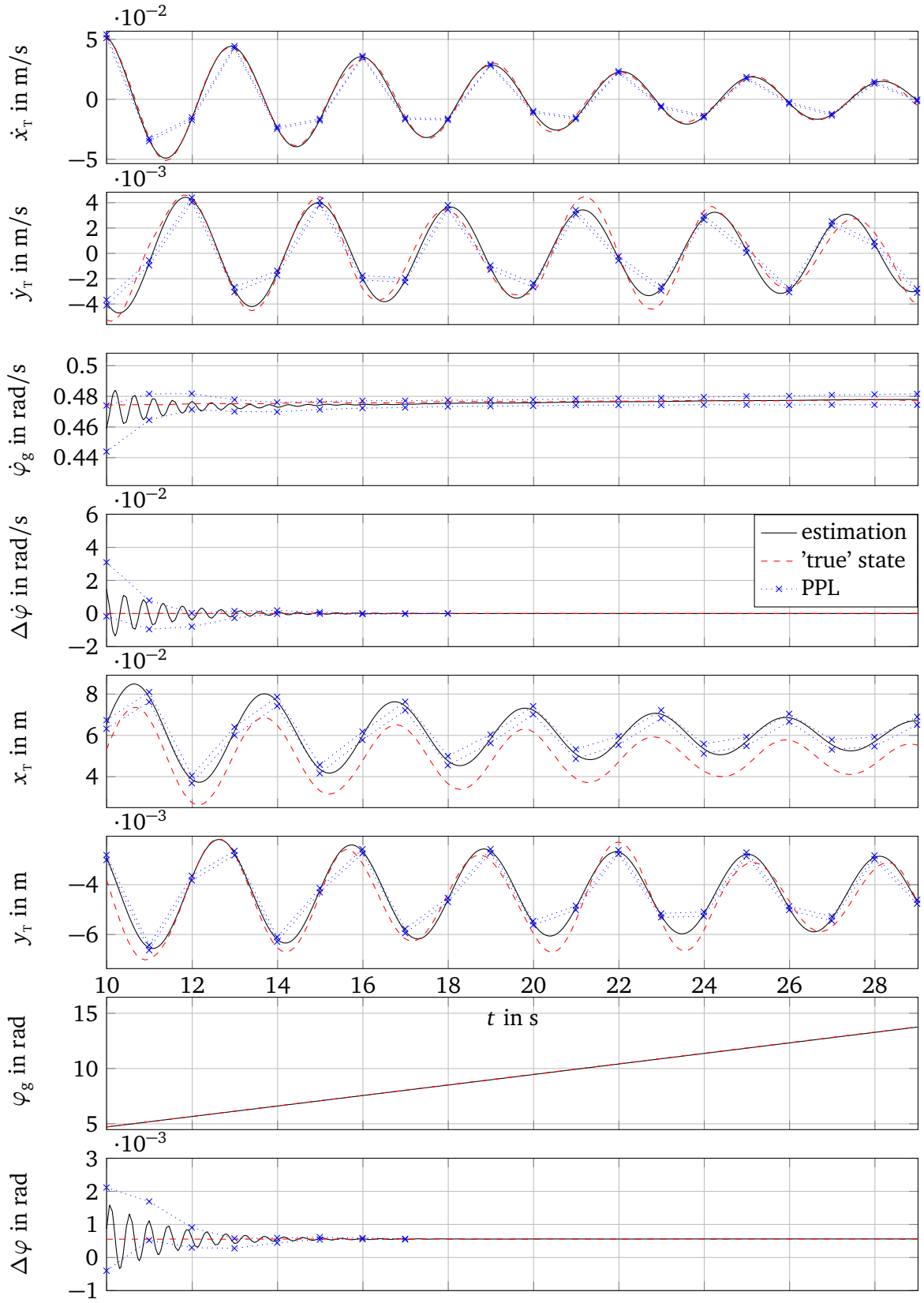

Figure 4.22.: Comparison of the 'true' states and their estimations including the PPL.
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `steady`, i. e. the wind is unknown.

4.5 Methodology Validation

In this section, we try to validate the application of the profile likelihood methodology to our use case. Therefore, we first evaluate the correlation between the width of the PL confidence intervals and the estimations errors for parameters. Then, we assess whether the PPL confidence intervals are a suitable measure for a state's observability.

4.5.1 Confidence Interval Width vs. Estimation Error

The statistical key aspect of the profile likelihood is the derivation of confidence intervals for parameter estimations from the χ^2-distribution. Assuming only Gaussian measurement noise, we can determine boundaries between which the true parameter value lies to a specified degree of certainty, e. g. 95 %. Concluding, the narrower the CI is, the better our estimation should be. In this section, we examine the correlation between these two factors. We do so by calculating the *Pearson correlation coefficient*

$$r = \frac{\sum_{i=1}^{n} \left(x_i - \overline{x}\right)\left(y_i - \overline{y}\right)}{\sqrt{\sum_{i=1}^{n} \left(x_i - \overline{x}\right)^2} \sqrt{\sum_{i=1}^{n} \left(y_i - \overline{y}\right)^2}}$$

with n being the sample size, x_i and y_i the single samples indexed with i and $\overline{x}$ and $\overline{y}$ the sample means.

Then, r indicates how strong the random variables x and y correlate, whereby it is always $-1 < r < 1$. For $r = 0$, the variables are completely uncorrelated, i. e. $P(x|y) = P(x)$ and $P(y|x) = P(y)$. For $r = 1$ or $r = -1$, both variables correlate perfectly linear, i. e. every pair $\left(x_i, y_i\right)$ lies on a straight line. The magnitude of its slope m is arbitrary, but it is $m > 0$ for $r > 0$ and $m < 0$ for $r < 0$.

In our case, the random variables x and y are the absolute confidence interval width of an unknown parameter and its absolute estimation error, i. e. regardless whether the estimation was too high or too low. Figure 4.23 shows the correlation for all free parameters from the simulations in Section 4.2 which were at least partly practically identifiable. In addition, the regression line is plotted. The pearson correlation coefficient is $r = 0.79$. Statistical tables exist to assess whether this correlation is statistically significant or not.[1] In our case, we have a sample size of $n = 88$ and $r > 0.283$ is sufficient to accept the hypothesis of a correlation between both parameters to a significance level of $\alpha = 0.01$.

Please note that the interpretation of the Pearson correlation coefficient is not always trivial. As [41] points out, a strong correlation between two variables does not necessarily mean that there is a cause-and-effect relationship. However, since the purpose of the profile likelihood

[1] Here, the table from `https://researchbasics.education.uconn.edu/r_critical_value_table/` was used.

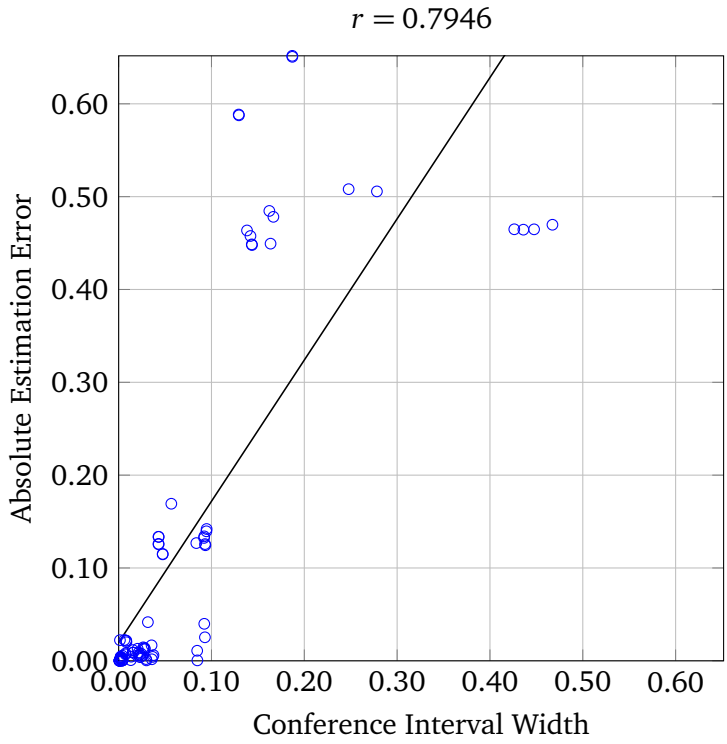

Figure 4.23.: Correlation of confidence interval widths and estimation errors for all constellations and all at least partly practically identifiable free parameters.

methodology is to determine statistical confidence intervals for a parameter estimation, we can expect this relation to be valid if our results suggest it.
He further suggests using the *coefficient of determination* r^2, which states how much of a percentage a change in y can be explained from changes in x. Here, that would be $0.79^2 = 62.41\,\%$ which would still indicate an existing relationship between the confidence interval's width and the estimation's error.

The correlation r varies if we consider only constellations with data sources from the model itself or not. Table 4.21 gives an overview of the resulting values. As can be seen, the correlation coefficient is the highest if only constellations are respected in which the data source was the model itself. This seems logical since the profile likelihood methodology assumes only measurement noise but no modeling error. However, even when only constellations with data from the advanced model are considered, r is still slightly higher than if all are considered, i. e. the data points could be better described by a single line. In all cases, the correlation was significant to a degree of at least $\alpha = 0.01$.

Concluding, the correlation between the confidence intervals' widths and the estimation errors of free parameters may not be linear. But, in any constellation, to a certainty of 99 % we can state that a wider confidence interval leads to a worse fit.

Table 4.21.: Correlation coefficients in dependence of considered data sources

Data Source	r	r^2	n	p
all	0.7946	0.6313	88	< 0.01
`'Basic_3Z'` or `'Standard_8Z'`	0.9510	0.9044	47	< 0.01
`'Advanced_21Z'`	0.8103	0.6565	41	< 0.01

4.5.2 PPL vs. State Estimation Error

Next, we validate our results regarding the systems' observability. Thus, we compare the PPL confidence intervals with the error of the estimated states. The confidence intervals of the initial values do not directly yield additional information, since we did also calculate the PPL for t_0, i. e. the confidence intervals are the same.

Figure 4.24 shows the results for the same constellation as in Figure 4.20, i. e. the basic model simulated with data from the advanced model and the wind being known. However, instead of the states, the states' errors are plotted, i. e. the black line represents the states' absolute estimation error, the dashed red line its mean value and the blue crosses again refer to the PPL, but here their half width is plotted. Plotting only the half width of each confidence interval is not entirely correct, since the upper and lower bound can have different distances to the estimated state value, but it still represents the expected error better.

As can be seen, the derived PPL confidence interval widths are mostly too small. Especially for the sinusoidal oscillations of the tower velocity and position, the PPL has problems to follow these oscillations (and is too small in general). For the angular speed, the PPL intervals decrease together with the estimation errors but stay bigger than the errors after $t = 20\,\mathrm{s}$.

Recalling the initial values' confidence intervals from Figure 4.15, we assumed solely the tower velocity to be only partly practically observable. However, the confidence intervals of the tower position was relatively wide, too. For all three initial values, the true values were not inside the confidence intervals. This could be interpreted as an indication for weak match between the PPL and the actual error.

Next, we analyze the same constellation for the standard model, i. e. as in Figure 4.18. The results for the errors are shown in Figure 4.25. Again, the PPL has the worst agreement with the oscillating states, i. e. the tower velocities and positions (plots 1, 2, 7, and 8). However, for the angular states, the PPL confidence intervals nearly perfectly fit the additional high-frequent oscillations in the transient phase (plots 3, 4, and 5). (As already mentioned in Section 4.4.1, the azimuth angle (plot 6) is measured and the errors are too small to derive any PPL intervals.) Considering the confidence intervals for the states' initial values in Figure 4.17 on page 85, we see that none of the tower velocities' and positions' confidence intervals contain the true value,

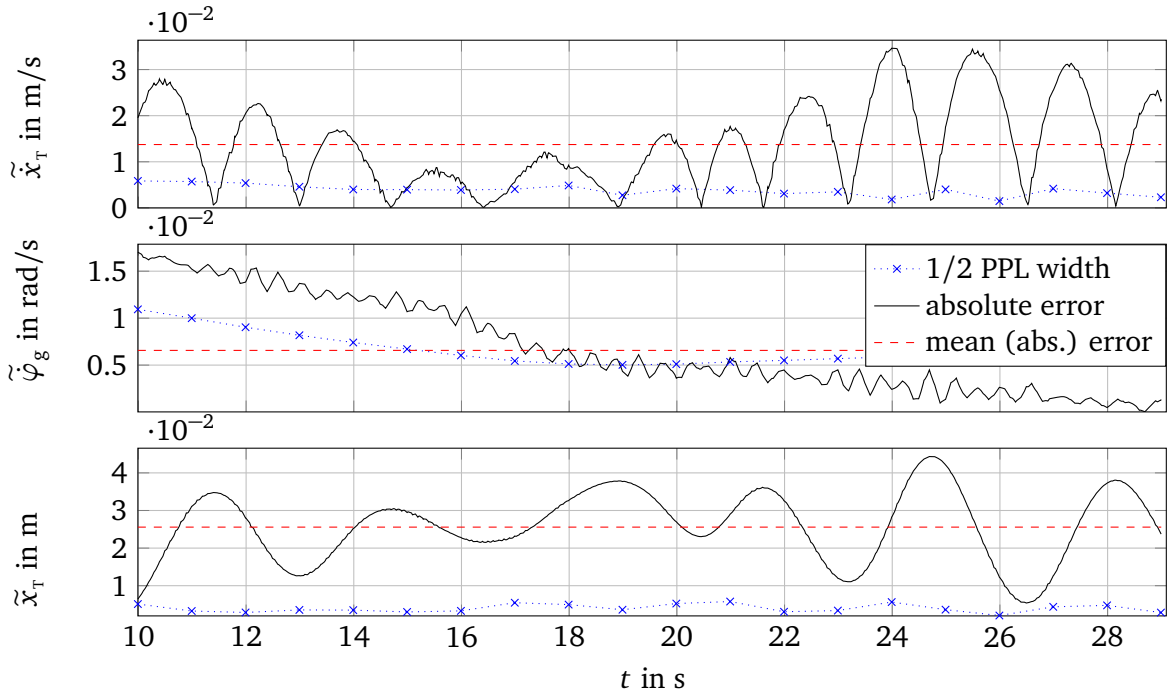

Figure 4.24.: Comparison of the PPL and the state estimations' errors.
Model: `Basic_3Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`, i. e. the wind is known.

but all of the angular quantities do. Hence, like for the basic model, the fitting of a states' initial value seems to be an indicator for the validity of the PPL.

However, we must consider that the statistical methodology of the PPL just as for the PL assumes only Gaussian measurement errors and therefore cannot be completely valid for modeling errors as we had here. Thus, we additionally simulated the basic model with data from itself. The results are given in Figure 4.26.

As can be seen, all errors are extremely small. The ripples show that at this magnitude, numerical issues become relevant. Otherwise, the PPL confidence intervals are very close to the errors until the oscillations become too strong. These oscillations of the error are due to either a bad estimated amplitude or a time shift, i. e. if the estimation cannot follow the true state properly.

Concluding, the PPL should not be blindly trusted. Especially when a state is mostly oscillating, its correct estimation is difficult and the resulting oscillating error mostly not sufficiently reflected by the PPL. The accuracy of a state's initial value's fitting seems to be an indicator how well the PPL can be trusted. However, this might also be only true for the angular quantities of our models. When evaluating the PPL intervals in this section, one should always keep in mind that most of the states' estimations are very well and the errors small. Hence, the PPL might be more accurate if measurement errors increase and modeling errors decrease.

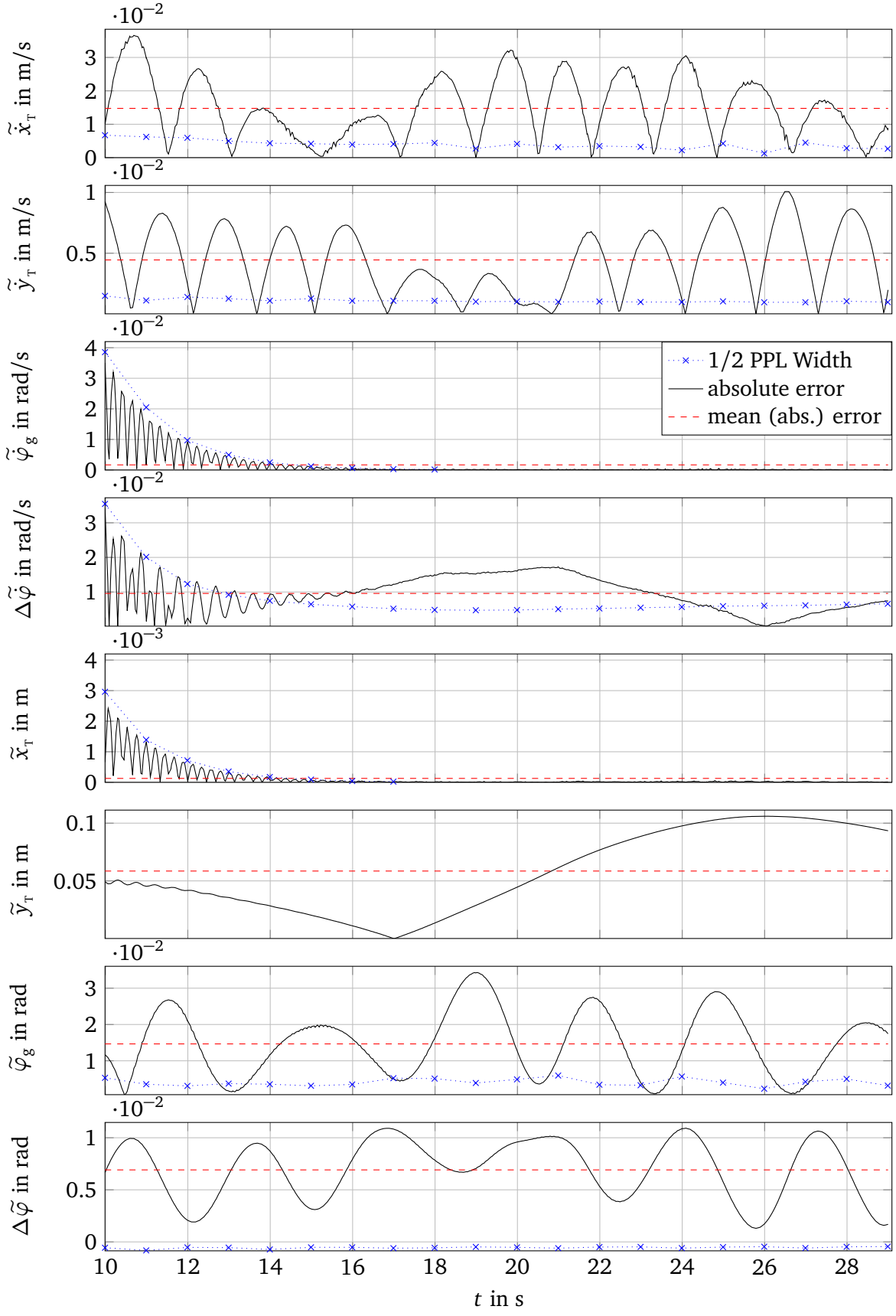

Figure 4.25.: Comparison of the PPL and the state estimations' errors.
Model: `Standard_8Z`, data source: `Advanced_21Z`, wind scenario: `high_turb`, i. e. the wind is known.

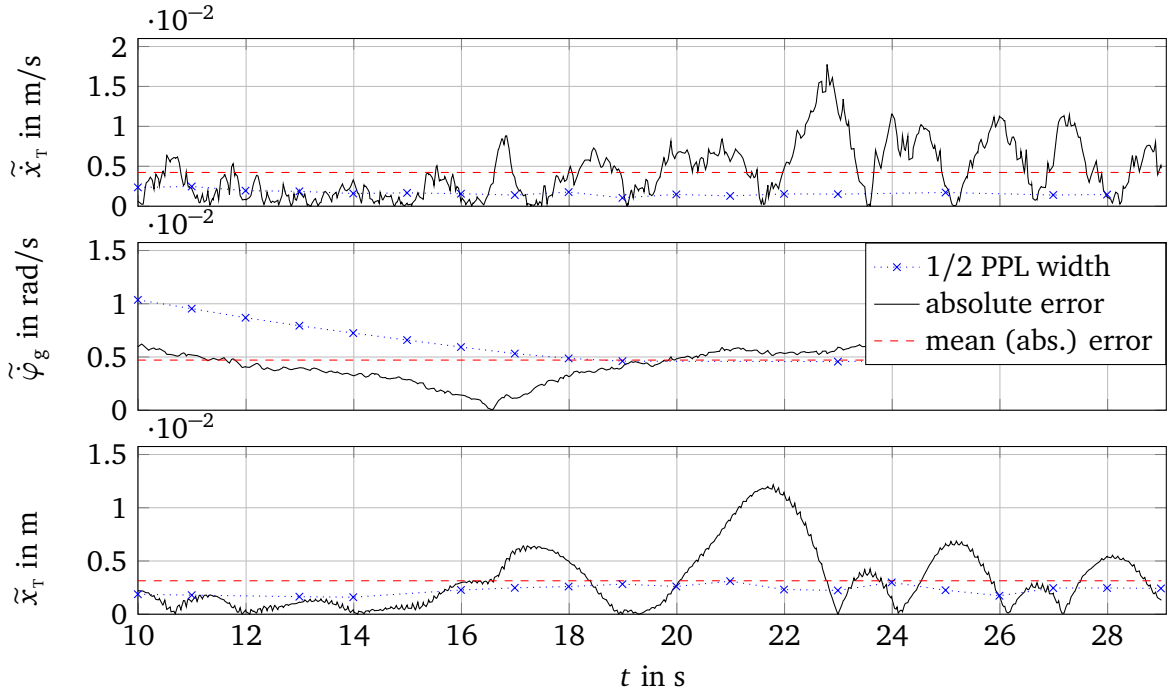

Figure 4.26.: Comparison of the PPL and the state estimations' errors.
Model: `Basic_3Z`, data source: `Basic_3Z`, wind scenario: `high_turb`, i. e. the wind is known.

5 Conclusions and Recommendations

In this last chapter, we first summarize the work which has been conducted. Secondly, we draw conclusions from it. Then, we end up with an outlook and possible future work which could build upon this one.

5.1 Summary

This thesis' aim was to find and evaluate a suitable method to assess the identifiability and observability of nonlinear systems and apply it to two wind turbine models. Thus, basic definitions and concepts of both identifiability and observability of nonlinear systems in general are discussed first, together with an interpretation of their relationship. Generally, both concepts are very close and could be substituted by the other, but only with drawbacks.

A thorough literature review revealed multiple methods for both concepts which have been discussed. Furthermore, for the first time a categorization by their purpose, i. e. whether the method itself is analytical or empirical and whether the results are of a qualitative or quantitative form has been carried out. Of all these methods, the profile likelihood methodology has been chosen, since it is conducted empirically by simulation but provides both qualitative as well as quantitative results. To the author's knowledge, this is the first time that this approach has been used for control engineering purposes. It consists of two methods, the profile likelihood (PL) and prediction profile likelihood (PPL) approach to assess a system's identifiability and observability, respectively. The basic methodology is that assuming only Gaussian distributed measurement noise, the log-likelihood for a vector $\boldsymbol{\theta}$ of parameter estimates is proportional to a weighted sum of squared residuals of this vector, $\chi^2_{\mathrm{PL}}(\boldsymbol{\theta})$. By optimizing this cost function for various values of a single parameter θ_i and comparing it to the commonly known χ^2-distribution, a confidence interval for this parameter can be derived. The parameter's identifiability then depends on the progression of the cost function and the size of this confidence interval. The PPL method transfers this approach to the uncertainty of a state estimation.

As part of our methodology, two nonlinear wind turbine models of different complexity are introduced. The less complex model, called the basic model, consists of three states, i. e. the fore-aft tower velocity, position and the angular generator speed. It is described by two ODEs. The more complex standard model consists of eight states. Additionally to the states from the basic model, the side-side tower velocity and position, the generator azimuth angle and the drive-train angular speed and torsion are respected, which leads to a total of four ODEs.

To apply the PL and PPL methods to our models, we use an open source MATLAB framework called Data2Dynamics (D2D). The models' equations are implemented in the D2D framework.

Input and measurement data is provided from previous simulations. Then, the models can be simulated with the D2D framework itself. Thereby, the air density and eigenfrequency can be treated as unknown parameters and estimated along the simulation. The wind velocity is either given as an input or identified, too. If so, a scenario with a constant wind velocity is used, since the profile likelihood methodology can neither handle time-varying parameters nor states with unknown dynamics. Furthermore, additional scenario factors are varied, such as the measurement noise, the measured outputs, the origin of the measurement data, the amount of measurement data and the wind turbulence. As origin of the measurement data, we do not only use data from the simulated model itself, but also from a more complex model with 21 states, called the advanced model. We assume this model's data to represent real-world data close enough to assess the model's identifiability and observability in a real-world application scenario.

The greater part of this thesis consists of the presentation of the simulation results. First, the influence of the data quality has been evaluated to decrease the total number of test scenarios. Therefore, a qualitative overview for both the identifiability and observability of both systems is given in dependence of which parameters are identified, if the wind velocity is known or not, which outputs are measured and which model is simulated with which data source. Thereby, we distinguish between structural and practical identifiability. Both concepts are introduced with the PL approach. Furthermore, we introduce and formalize similar concepts of structural and practical observability in dependence of the identifiability of a state's initial value.

After the qualitative overview, we first analyze the identifiability of the most important combinations in detail. Thereby, the accuracy of the parameters' estimations as well as their derived confidence intervals are considered. Then, we further analyze the observability of both systems if all outputs are measured. Thus, the PPL method is applied and all states are estimated.

Finally, we validated the use of the profile likelihood methodology for our use case. For the identifiability assessment, the Pearson correlation of the parameters' confidence intervals' width and their estimation errors is calculated. For the observability assessment, we compare the derived PPL confidence intervals of the state estimations with their estimation errors.

5.2 Conclusions

In this work, the identifiability and observability of two nonlinear wind turbine models has successfully been analyzed by using the profile likelihood methodology. It represents the best compromise between flexibility and expressiveness of the results. That is, it can be used with any kind of empirical data, e. g. with data from the model itself or with real measurement data. Furthermore, it assesses the model's identifiability and observability both qualitatively and quantitatively, i. e. it can be distinguished between structural and practical identifiability. This concept has also formally been extended to structural and practical observability of a state. However, the validity of identifiability and observability assessment with the profile likelihood

methodology is always constrained to the input scenario that has been tested, i. e. it does not provide analytical conditions for the model's identifiability or observability.

The implementation of both models in the open source MATLAB framework D2D is automated so that the user can simulate either system and easily choose from various scenarios, such as the data source, the wind scenario, the number of data points to be used, the parameters which are to be estimated, the measured outputs and more. Since the simulations are computationally demanding especially for the more complex standard model with eight states, the influence of the data quality has been evaluated a priori. As a conclusion, the simulations have been performed with $n = 200$ data points which had a time interval of $\Delta T = 0.1\,\mathrm{s}$ and artificial measurement noise with a 30 dB signal-to-noise ratio. Our results showed, that for both models, all three parameters are well identifiable by themselves if all outputs are measured. However, if the air density and the wind velocity are identified together, their effects cannot be distinguished well enough. Therefore, they are structural but not practically identifiable. This is independent of whether the eigenfrequency is identified additionally or not.
The analysis of the models' observability showed that even if the air density and wind velocity cannot be properly distinguished, their combined effect is generally estimated well enough to observe the states. In the transient phase, the angular quantities' estimations show more and stronger oscillations than the 'true' states from the more complex model. Furthermore, the tower positions are estimated with a small offset. However, both models are well enough observable for a real-world application.

Even though the D2D framework does not provide any algorithms to identify parameters or observe states online, its state estimations which come along with the simulations can be considered as a benchmark for observers which are to be designed. That is, the accuracy of its estimations can be considered as a degree of quality an observer can at least achieve.

Furthermore, our validation showed that the correlation of a parameter's confidence interval obtained by the PL and the accuracy of its estimation is statistically significant, i. e. we can compare two parameters' identifiability with each other, even if the true value is not known. However, for our use case with a dominating modeling error, the PPL cannot be blindly trusted. Especially for oscillating states as the tower velocities and positions, the PPL confidence intervals are usually estimated to narrow, i. e. the framework is to confident in its estimations. For the angular quantities, the PPL confidence intervals fit their errors in the transient phase accurately.

5.3 Outlook and Future Work

The analysis of a wind turbine's identifiability and observability in this work is limited to some constraints. First, only two models with reduced complexity have been analyzed. If in practice the wind turbine shall be represented by a more complex model, that model needs to be analyzed, too. Depending on the model equations, the D2D framework might reach its limits. However, since it is an open source project, additional functionality could be implemented.

Secondly, the standard model has only been simulated in the partial load regime. For higher wind velocities, the aerodynamic characteristics need to be realized in the D2D framework. To this date, a function `LUT_bilinear` is implemented but unstable for grids greater than 4×4 points.
Thirdly, data from an advanced model with 21 states has been used to analyze the models' identifiability and observability in a real-world environment. To further validate the results, data from the FAST simulator or real on-site measurement data could be used.
Furthermore, to reduce computation time, a cluster of MATLAB workers could be used.[1]

Despite the above noted limitations, our results showed that both models are mostly identifiable as well as observable, at least if all outputs are measured. Therefore, an appropriate observer can now be implemented. However, a nontrivial problem is the estimation of the wind velocity, since it is highly time-variant like a state but has unknown dynamics. Furthermore, e. g. the eigenfrequency does only slowly vary over time. Hence, its identification could be outsourced from the regular system observation and be proceeded by an external algorithm, e. g. the profile likelihood approach. A similar procedure could also be thought of for the wind velocity. However, since the wind velocity changes for every discrete point in time, the computational time could be too high.

[1] See `https://github.com/Data2Dynamics/d2d/wiki/Distributed-Computing` for more information.

A Appendix

Listing A.1: *model.def* file for the basic model with three states. The wind velocity is considered as a known input, but the eigenfrequency and air density as unknown parameters.

```
DESCRIPTION
"Automatisch erstelltes .def-file"
PREDICTOR
t T s time 0.000000 1.000000

COMPARTMENTS

STATES
QD_GeAz C    "nM" "conc." default 1 "Zust." 0
NcIMUTVxs C  "nM" "conc." default 1 "Zust." 0
TTDspFA C    "nM" "conc." default 1 "Zust." 0

INPUTS
GenTq C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [16060.1436, →
    ←16056.2756, 16026.9262, 16026.9262])"
Wind1VelX C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [7.2832, 7.3199, →
    ←7.2582, 7.2582])"

ODES
"(1.225*AirDensity)/2*PI*BldTpRd^3 /(HubInertia+BldInertia1+BldInertia2+→
    ←BldInertia3+GenInertia*GbRatio^2)*  (cm2*(QD_GeAz * BldTpRd / (Wind1VelX - →
    ←NcIMUTVxs))^2 + cm1*(QD_GeAz * BldTpRd / (Wind1VelX - NcIMUTVxs)) + cm0)* (→
    ←Wind1VelX - NcIMUTVxs)^2   - GbRatio * GenTq/ (HubInertia+BldInertia1+→
    ←BldInertia2+BldInertia3+GenInertia*GbRatio^2)"
"(1.225*AirDensity)/2*PI*BldTpRd^2 /EqTwrMass * (ct2*(QD_GeAz * BldTpRd / (→
    ←Wind1VelX - NcIMUTVxs))^2  + ct1*(QD_GeAz * BldTpRd / (Wind1VelX - NcIMUTVxs)→
    ←) + ct0) * (Wind1VelX - NcIMUTVxs)^2   - 2* TwrDmpFA / (2 * EqTwrMass * →
    ←(2.0647*Omega0X)) * (2.0647*Omega0X) * NcIMUTVxs - (2.0647*Omega0X)^2 * →
    ←TTDspFA"
"NcIMUTVxs"

DERIVED

OBSERVABLES
y1_GenSpeed C "au" "conc." 0 0 "60/(2*PI) * GbRatio *QD_GeAz"
y2_NcIMUTAxs C "au" "conc." 0 0 "(1.225*AirDensity)/2*PI*BldTpRd^2 /EqTwrMass * (→
    ←ct2*(QD_GeAz * BldTpRd / (Wind1VelX - NcIMUTVxs))^2  + ct1*(QD_GeAz * BldTpRd→
    ← / (Wind1VelX - NcIMUTVxs)) + ct0) * (Wind1VelX - NcIMUTVxs)^2   - 2* →
```

```
    ←TwrDmpFA / (2 * EqTwrMass * (2.0647*Omega0X)) * (2.0647*Omega0X) * NcIMUTVxs →
    ←- (2.0647*Omega0X)^2 * TTDspFA"

ERRORS
y1_GenSpeed  "GenSpeed_abs"
y2_NcIMUTAxs     "NcIMUTAxs_abs"

CONDITIONS
BldTpRd      "63"
GbRatio      "97"
TwrDmpFA     "9284.9723"
EqTwrMass    "449710.2273"
GenInertia   "534.116"
HubInertia   "115926"
BldInertia1      "11776047"
BldInertia2      "11776047"
BldInertia3      "11776047"
PI   "3.1416"
cm2      "-0.00032089"
cm1      "-0.0025954"
cm0      "0.10059"
ct2      "-0.0064661"
ct1      "0.15986"
ct0      "-0.082925"
```

Listing A.2: *model.def* file for the standard model with eight states. The wind velocity is considered as a known input, but the eigenfrequency and air density as unknown parameters.

```
DESCRIPTION
"Automatisch erstelltes .def-file"
PREDICTOR
t T s time 0.000000 1.000000

COMPARTMENTS

STATES
NcIMUTVxs C  "nM" "conc." default 1 "Zust." 0
NcIMUTVys C  "nM" "conc." default 1 "Zust." 0
QD_GeAz C    "nM" "conc." default 1 "Zust." 0
QD_DrTr C    "nM" "conc." default 1 "Zust." 0
TTDspFA C    "nM" "conc." default 1 "Zust." 0
TTDspSS C    "nM" "conc." default 1 "Zust." 0
Q_GeAz C     "nM" "conc." default 1 "Zust." 0
Q_DrTr C     "nM" "conc." default 1 "Zust." 0

INPUTS
GenTq C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [16060.1436, →
    ←16056.2756, 16026.9262, 16026.9262])"
```

```
BldPitch1 C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [0, 0, 0, 0])"
BldPitch2 C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [0, 0, 0, 0])"
BldPitch3 C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [0, 0, 0, 0])"
Wind1VelX C "au" "conc." "inputspline(t, 3, [0, 0.1, 0.2, 0.3], [7.2832, 7.3199, 7.2582, 7.2582])"

ODES
" -2*zetaX*(2.0647*Omega0X)*NcIMUTVxs -((2.0647*Omega0X)^2)*TTDspFA + 1/EqTwrMass * ((1.225*AirDensity)/6 * PI*BldTpRd^2 * (( 1+BeamCoup1 * NormEffRd1 * cos((Q_GeAz + Q_DrTr))) * (ct2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs )))^2 + ct1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs ))) +ct0)*( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs )^2 +( 1+BeamCoup1 * NormEffRd1 * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * (ct2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )))^2 + ct1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs ))) +ct0)*( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )^2 +( 1+BeamCoup1 * NormEffRd1 * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * (ct2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )))^2 + ct1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs ))) +ct0)*( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )^2)) "
"-2*zetaY*(2.0647*Omega0X)*NcIMUTVys - ((2.0647*Omega0X)^2)*TTDspSS -1/EqTwrMass * ( (1.225*AirDensity)/6 * PI*BldTpRd^3 /TangEffRd1 * ( cos((Q_GeAz + Q_DrTr)) * (cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs )))^2 + cm1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs ))) +cm0) * ( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs )^2 + cos((Q_GeAz + Q_DrTr + 2 * PI / 3))* (cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs ))^2 + cm1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd
```

```
    ←* cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs ))) +cm0) * ( (1+1/→
    ←HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX →
    ←- (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )→
    ←^2 + cos((Q_GeAz + Q_DrTr + 4 * PI / 3))* (cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd→
    ←*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3)))^VerShr →
    ←*Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * →
    ←NcIMUTVxs )))^2 + cm1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *→
    ←PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ →
    ←BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs ))) +→
    ←cm0) * ( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3)))^→
    ←VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / →
    ←3))) * NcIMUTVxs )^2 )+ BeamCoup1*GbRatio*GenTq )"
"1/(GenInertia*GbRatio^2)*(-GbRatio*GenTq + DtTorDmp*QD_DrTr + DtTorStfn*Q_DrTr)"
"1/(GenInertia*GbRatio^2) *GbRatio*GenTq -(1/ (HubInertia+BldInertia1+BldInertia2→
    ←+BldInertia3+GenInertia*GbRatio^2) + 1/(GenInertia*GbRatio^2)) * DtTorDmp*→
    ←QD_DrTr -(1/ (HubInertia+BldInertia1+BldInertia2+BldInertia3+GenInertia*→
    ←GbRatio^2) + 1/(GenInertia*GbRatio^2)) * DtTorStfn*Q_DrTr + 1/ (HubInertia+→
    ←BldInertia1+BldInertia2+BldInertia3+GenInertia*GbRatio^2)*(1.225*AirDensity)→
    ←/6 * PI*BldTpRd^3 * ((cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight →
    ←*PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*→
    ←PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs )))^2 + cm1 * ((QD_GeAz + →
    ←QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^→
    ←VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * →
    ←NcIMUTVxs ))) +cm0) * ( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^→
    ←VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * →
    ←NcIMUTVxs )^2 + (cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *→
    ←PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ →
    ←BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )))^2 +→
    ← cm1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((→
    ←Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * →
    ←cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs ))) +cm0)*( (1+1/HubHeight *→
    ←PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ →
    ←BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )^2 +(→
    ←cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((→
    ←Q_GeAz + Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * →
    ←cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )))^2 + cm1 * ((QD_GeAz + →
    ←QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * →
    ←PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr +→
    ← 4 * PI / 3))) * NcIMUTVxs ))) +cm0)*( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz→
    ← + Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((→
    ←Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )^2 )"
"NcIMUTVxs"
"NcIMUTVys"
"QD_GeAz"
"QD_DrTr"

DERIVED
```

```
37 OBSERVABLES
   y1_GenSpeed C "au" "conc." 0 0 "60/(2*PI) * GbRatio *QD_GeAz"
   y2_NcIMUTAxs C "au" "conc." 0 0 " -2*zetaX*(2.0647*Omega0X)*NcIMUTVxs -((2.0647*→
       ←Omega0X)^2)*TTDspFA + 1/EqTwrMass * ((1.225*AirDensity)/6 * PI*BldTpRd^2 * ((→
       ← 1+BeamCoup1 * NormEffRd1 * cos((Q_GeAz + Q_DrTr))) * (ct2 * ((QD_GeAz + →
       ←QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^→
       ←VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * →
       ←NcIMUTVxs )))^2 + ct1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *→
       ←PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd→
       ← * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs ))) +ct0)*( (1+1/HubHeight *PwrEffRd *→
       ← cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((→
       ←Q_GeAz + Q_DrTr))) * NcIMUTVxs )^2 +( 1+BeamCoup1 * NormEffRd1 * cos((Q_GeAz →
       ←+ Q_DrTr + 2 * PI / 3))) * (ct2 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/→
       ←HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX →
       ←- (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )→
       ←))^2 + ct1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos→
       ←((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd →
       ←* cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs ))) +ct0)*( (1+1/HubHeight→
       ← *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *Wind1VelX - (1+ →
       ←BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * NcIMUTVxs )^2 +( →
       ←1+BeamCoup1 * NormEffRd1 * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))) * (ct2 * ((→
       ←QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + →
       ←Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((→
       ←Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )))^2 + ct1 * ((QD_GeAz + QD_DrTr→
       ←)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))→
       ←)^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI →
       ←/ 3))) * NcIMUTVxs ))) +ct0)*( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + →
       ←Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((→
       ←Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )^2)) "
   y3_Azimuth C "au" "conc." 0 0 "57.3*(Q_GeAz + Q_DrTr)"
   y4_NcIMUTAys C "au" "conc." 0 0 "-2*zetaY*(2.0647*Omega0X)*NcIMUTVys - ((2.0647*→
       ←Omega0X)^2)*TTDspSS -1/EqTwrMass * ( (1.225*AirDensity)/6 * PI*BldTpRd^3 /→
       ←TangEffRd1 * ( cos((Q_GeAz + Q_DrTr))* (cm2 * ((QD_GeAz + QD_DrTr)*BldTpRd→
       ←*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr)))^VerShr *Wind1VelX - →
       ←(1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * NcIMUTVxs )))^2 + cm1 * ((→
       ←QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + →
       ←Q_DrTr)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))→
       ←) * NcIMUTVxs ))) +cm0) * ( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr))→
       ←)^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr))) * →
       ←NcIMUTVxs )^2 + cos((Q_GeAz + Q_DrTr + 2 * PI / 3))* (cm2 * ((QD_GeAz + →
       ←QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * →
       ←PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr +→
       ← 2 * PI / 3))) * NcIMUTVxs )))^2 + cm1 * ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( →
       ←(1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3)))^VerShr *→
       ←Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 * PI / 3))) * →
       ←NcIMUTVxs ))) +cm0) * ( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 2 *→
       ← PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr →
       ←+ 2 * PI / 3))) * NcIMUTVxs )^2 + cos((Q_GeAz + Q_DrTr + 4 * PI / 3))* (cm2 *→
```

```
    ← ((QD_GeAz + QD_DrTr)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + →
    ←Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((→
    ←Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )))^2 + cm1 * ((QD_GeAz + QD_DrTr→
    ←)*BldTpRd*(1/( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI / 3))→
    ←)^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((Q_GeAz + Q_DrTr + 4 * PI →
    ←/ 3))) * NcIMUTVxs ))) +cm0) * ( (1+1/HubHeight *PwrEffRd * cos((Q_GeAz + →
    ←Q_DrTr + 4 * PI / 3)))^VerShr *Wind1VelX - (1+ BeamCoup1*PwrEffRd * cos((→
    ←Q_GeAz + Q_DrTr + 4 * PI / 3))) * NcIMUTVxs )^2 )+ BeamCoup1*GbRatio*GenTq )"
42
   ERRORS
   y1_GenSpeed  "GenSpeed_abs"
   y2_NcIMUTAxs     "NcIMUTAxs_abs"
   y3_Azimuth   "Azimuth_abs"
47 y4_NcIMUTAys     "NcIMUTAys_abs"

   CONDITIONS
   BldTpRd      "63"
   TangEffRd1   "31.5"
52 NormEffRd1   "31.5"
   PwrEffRd     "31.5"
   HubHeight    "87.6"
   BeamCoup1    "0.017123"
   GbRatio      "97"
57 DtTorStfn    "867637000"
   DtTorDmp     "6215000"
   EqTwrMass    "449710.2273"
   GenInertia   "534.116"
   HubInertia   "115926"
62 BldInertia1      "11776047"
   BldInertia2      "11776047"
   BldInertia3      "11776047"
   VerShr   "0"
   PI   "3.1416"
67 cm2      "-0.00032089"
   cm1      "-0.0025954"
   cm0      "0.10059"
   ct2      "-0.0064661"
   ct1      "0.15986"
72 ct0      "-0.082925"
   zetaX    "0.005"
   zetaY    "0.0048888"
```

Bibliography

[1] Manwell, James F., Jon G. McGowan and Anthony L. Rogers: *Wind energy explained: Theory, Design and Application*. John Wiley & Sons, 2010.

[2] Miao, Hongyu, Xiaohua Xia, Alan S. Perelson and Hulin Wu: *On Identifiability of Nonlinear ODE Models and Applications in Viral Dynamics*. SIAM Review, 53(1):3–39, 2011.

[3] Ljung, Lennart: *Convergence analysis of parametric identification methods*. IEEE transactions on automatic control, 23(5):770–783, 1978.

[4] Ljung, Lennart and Torkel Glad: *On global identifiability for arbitrary model parametrizations*. Automatica, 30(2):265 – 276, 1994.

[5] Raue, Andreas, Johan Karlsson, Maria Pia Saccomani, Mats Jirstrand and Jens Timmer: *Comparison of approaches for parameter identifiability analysis of biological systems*. Bioinformatics, 30(10):1440, 2014.

[6] Sontag, Eduardo D.: *For Differential Equations with r Parameters, 2r+1 Experiments Are Enough for Identification*. Journal of Nonlinear Science, 12(6):553–583, Jan 2003.

[7] Thomaseth, Karl and Claudio Cobelli: *Generalized sensitivity functions in physiological system identification*. Annals of biomedical engineering, 27(5):607–616, 1999.

[8] Adamy, Jürgen: *Nichtlineare Systeme und Regelungen*. Berlin, Heidelberg, 2nd, rev. and ext. edition, 2014.

[9] Hermann, Robert and Arthur Krener: *Nonlinear controllability and observability*. IEEE Transactions on automatic control, 22(5):728–740, 1977.

[10] Sedoglavic, Alexandre: *A Probabilistic Algorithm to Test Local Algebraic Observability in Polynomial Time*. In *Proceedings of the 2001 International Symposium on Symbolic and Algebraic Computation*, ISSAC '01, pages 309–317, New York, NY, USA, 2001. ACM.

[11] Krener, Arthur J. and Kayo Ide: *Measures of unobservability*. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 6401–6406, Dec 2009.

[12] Raue, A., V. Becker, U. Klingmüller and J. Timmer: *Identifiability and observability analysis for experimental design in nonlinear dynamical models*. Chaos: An Interdisciplinary Journal of Nonlinear Science, 20(4):045105, 2010.

[13] Quaiser, Tom and Martin Mönnigmann: *Systematic identifiability testing for unambiguous mechanistic modeling – application to JAK-STAT, MAP kinase, and NF-κ B signaling pathway models*. BMC Systems Biology, 3(1):50, 2009.

[14] Ehlers, Julian, Amadou Diop and Henrik Bindner: *Sensor selection and state estimation for wind turbine controls*. In *45th AIAA Aerospace Sciences Meeting and Exhibit*, page 1019, 2007.

[15] Pohjanpalo, H: *System identifiability based on the power series expansion of the solution*. Mathematical biosciences, 41(1-2):21–33, 1978.

[16] Walter, Eric and Yves Lecourtier: *Unidentifiable compartmental models: What to do?* Mathematical biosciences, 56(1-2):1–25, 1981.

[17] Vajda, Sandor, Keith R Godfrey and Herschel Rabitz: *Similarity transformation approach to identifiability analysis of nonlinear compartmental models*. Mathematical biosciences, 93(2):217–248, 1989.

[18] Vajda, S and H Rabitz: *State isomorphism approach to global identifiability of nonlinear systems*. IEEE Transactions on Automatic Control, 34(2):220–223, 1989.

[19] Denis-Vidal, Lilianne and Ghislaine Joly-Blanchard: *An Easy to Check Criterion for (Un)identifiability of Uncontrolled Systems and Its Applications*. IEEE Transactions on Automatic Control, 45(4):768–771, 2000.

[20] Jeffrey, Annah M. and Xiaohua Xia: *Identifiability of HIV/AIDS models*. Deterministic and Stochastic Models of AIDS Epidemics and HIV Infections with Intervention, pages 255–286, 2005.

[21] Xia, Xiaohua and Claude H. Moog: *Identifiability of nonlinear systems with application to HIV/AIDS models*. IEEE transactions on automatic control, 48(2):330–336, 2003.

[22] Wu, Hulin, Haihong Zhu, Hongyu Miao and Alan S. Perelson: *Parameter identifiability and estimation of HIV/AIDS dynamic models*. Bulletin of mathematical biology, 70(3):785–799, 2008.

[23] Raue, A., C. Kreutz, T. Maiwald, J. Bachmann, M. Schilling, U. Klingmüller and J. Timmer: *Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood*. Bioinformatics, 25(15):1923, 2009.

[24] Hengl, S., C. Kreutz, J. Timmer and T. Maiwald: *Data-based identifiability analysis of non-linear dynamical models*. Bioinformatics, 23(19):2612, 2007.

[25] Kreutz, Clemens, Andreas Raue and Jens Timmer: *Likelihood based observability analysis and confidence intervals for predictions of dynamic models*. BMC Systems Biology, 6(1):120, 2012.

[26] Adamy, Jürgen: *Systemdynamik und Regelungstechnik II*. Aachen, 5th edition, 2016.

[27] Brammer, Karl and Gerhard Siffling: *Kalman-bucy filters*. Artech House on Demand, 1989.

[28] Birk, J. and M. Zeitz: *Computer-Aided Analysis of Nonlinear Observation Problems*. {IFAC} Proceedings Volumes, 25(13):257 – 262, 1992. 2nd {IFAC} Symposium on Nonlinear Control Systems Design 1992, Bordeaux, France, 24-26 June.

[29] Anguelova, Milena: *Nonlinear observability and identifiability: General theory and a case study of a kinetic model for S. cerevisiae*. 2004.

[30] Lall, Sanjay, Jerrold E. Marsden and Sonja Glavaški: *Empirical model reduction of controlled nonlinear systems*. International Federation of Automatic Control, 1999.

[31] Hahn, Juergen and Thomas F. Edgar: *An improved method for nonlinear model reduction using balancing of empirical gramians*. Computers & Chemical Engineering, 26(10):1379 – 1397, 2002.

[32] Geffen, D., R. Findeisen, M. Schliemann, Frank Allgöwer and M. Guay: *Observability based parameter identifiability for biochemical reaction networks*. In *American Control Conference, 2008*, pages 2130–2135. IEEE, 2008.

[33] Eberle, Claudia and Christoph Ament: *Real-time state estimation and long-term model adaptation: a two-sided approach toward personalized diagnosis of glucose and insulin levels*. Journal of diabetes science and technology, 6(5):1148–1158, 2012.

[34] Glotzbach, Thomas, Naveena Crasta and Christoph Ament: *Observability analyses and trajectory planning for tracking of an underwater robot using empirical Gramians*. IFAC Proceedings Volumes, 47(3):4215–4221, 2014.

[35] Himpe, Christian and Mario Ohlberger: *A unified software framework for empirical gramians*. Journal of Mathematics, 2013.

[36] Gil, Edwin C. M.: *Nonlinear Observability Analysis for Wind Turbines using Empirical Gramian Matrices*. 2017.

[37] Raue, A., B. Steiert, M. Schelker, C. Kreutz, T. Maiwald, H. Hass, J. Vanlier, C. Tönsing, L. Adlung, R. Engesser, W. Mader, T. Heinemann, J. Hasenauer, M. Schilling, T. Höfer, E. Klipp, F. Theis, U. Klingmüller, B. Schöberl and J. Timmer: *Data2Dynamics: a modeling environment tailored to parameter estimation in dynamical systems*. Bioinformatics, 31(21):3558, 2015.

[38] Vanlier, J., C. A. Tiemann, P. A. J. Hilbers and N. A. W. van Riel: *A Bayesian approach to targeted experiment design*. Bioinformatics, 28(8):1136, 2012.

[39] Ritter, B., A. Schild, M. Feldt and U. Konigorski: *The design of nonlinear observers for wind turbine dynamic state and parameter estimation*. In *Journal of Physics: Conference Series*, volume 753, page 052029. IOP Publishing, 2016.

[40] Raue, Andreas, Marcel Schilling, Julie Bachmann, Andrew Matteson, Max Schelke, Daniel Kaschek, Sabine Hug, Clemens Kreutz, Brian D. Harms, Fabian J. Theis, Ursula Klingmüller and Jens Timmer: *Lessons Learned from Quantitative Dynamical Modeling in Systems Biology*. PLOS ONE, 8(9):1–17, 09 2013.

[41] Taylor, Richard: *Interpretation of the correlation coefficient: a basic review*. Journal of diagnostic medical sonography, 6(1):35–39, 1990.